Let Us C Solutions

(2nd Revised Edition)

Yashavant P. Kanetkar

BPB PUBLICATIONS
B-14, CONNAUGHT PLACE, NEW DELHI-1

www.indianfun.org

www.indianfun.org

Contents

Chapter 1

Getting Started

[A] Which of the following are invalid variable names and why?

InterestPaid	: Valid
si-int	: Invalid. No special symbol other than underscore (_) can be used in a variable name.
AVERAGE	: Valid
percent	: Invalid. '.' is not allowed in a variable name.
123	: Invalid. Variable name must begin with an alphabet.
dist in km	: Invalid. Blanks are not allowed within a variable name.
ot pay	: Invalid. Blanks are not allowed within a variable name.
Name	: Valid
FLOAT	: Valid. Because **float** & FLOAT are different.

[B] Point out the Errors, if any, in the following C statements:

(a) int = 314.562 * 150 ;

Error. **int** is a keyword hence should not be used as a variable.

(b) name = 'Ajay' ;

Error. 'Ajay' is an invalid character constant.

(c) 3.14 * r * r = area ;

Error. On the left-hand side of equal to (=) there can only be a variable.

(d) k = a * b + c (2.5a + b) ;

Error. Multiplication operator (*) missing between variable **c** & the opening parenthesis (( ) and between **2.5** and variable **a**.

(e) m_inst = rate of interest * amount in rs ;

Error. **rate of interest** and **amount in rs** are invalid variable names.

(f) si = principal * rateofinterest * numberofyears / 100 ;

No Error

(g) area = 3.14 * r ** 2 ;

Error. ' ** ' is an invalid operator.

[C] Evaluate the following expressions and show their hierarchy:

(a) g = big / 2 + big * 4 / big - big + abc / 3 ;

(abc = 1.5, big = 3, assume **g** to be a **float**)

Answer:

g = **3 / 2** + 3 * 4 / 3 – 3 + 1.5 / 3	operation: /
g = 1 + **3 * 4** / 3 – 3 + 1.5 / 3	operation: *
g = 1 + **12 / 3** – 3 + 1.5 / 3	operation: /
g = 1 + 4 – 3 + **1.5 / 3**	operation: /
g = **1 + 4** – 3 + 0.5	operation: +
g = **5 – 3** + 0.5	operation: -
g = **2 + 0.5**	operation: +
g = 2.5	

(b) on = ink * act / 2 + 3 / 2 * act + 2 + tig ;

(ink = 3, act = 2, tig = 3.2, assume **on** to be an **int**)

Answer:

on = **3 * 2** / 2 + 3 / 2 * 2 + 2 + 3.2	operation: *
on = **6 / 2** + 3 / 2 * 2 + 2+ 3.2	operation: /
on = 3 + **3 / 2** * 2 + 2+ 3.2	operation: /
on = 3 + **1 * 2** + 2+ 3.2	operation: *
on = **3 + 2** + 2+ 3.2	operation: +
on = **5 + 2**+ 3.2	operation: +
on = **7+ 3.2**	operation: +
on = 10	

(c) s = qui * add / 4 - 6 / 2 + 2 / 3 * 6 / god ;

(qui = 2, add = 4, god = 3, assume **s** to be an **int**)

Answer:

s = **2 * 4** / 4 – 6 / 2 + 2 / 3 * 6 / 3	operation: *
s = **8 / 4** - 6 / 2 + 2 / 3 * 6 / 3	operation: /
s = 2 - **6 / 2** + 2 / 3 * 6 / 3	operation: /
s = 2 - 3 + **2 / 3** * 6 / 3	operation: /
s = 2 - 3 + **0 * 6** / 3	operation: *
s = 2 - 3 + **0 / 3**	operation: /
s = **2 - 3** + 0	operation: -
s = **-1 + 0**	operation: +
s = -1	

[D] Convert the following equations into corresponding C statements:

www.indianfun.org
www.indianfun.org

(a) $Z = \frac{8.8(a+b)2/c - 0.5 + 2a/(q+r)}{(a+b)*(l/m)}$

Answer:

```
Z=((8.8*(a+b)*2/c)-(0.5+2*a/(q+r)))/((a+b)
(1/m))
```

(b) $X = \frac{-b+(b.*b)+2-4ac}{2a}$

Answer:

```
X=(-b+(b*b)+2-4*a*c)/(2*a)
```

(c) $R = \frac{2v+6.22(c+d)}{g+v}$

Answer:

```
R=(2*v+6.22*(c+d))/(g+v)
```

[E] What would be the output of the following program segment?

```
int i = 2, j = 3, k, l ;
float a, b ;
k = i / j * j ;
l = j / i * i ;
a = i / j * j ;
b = j / i * i ;
printf ( "%d %d %f %f", k, l, a, b ) ;
```

Output:

```
0 2 0.000000 2.000000
```

[F] Pick up the correct alternative for each of the following questions:

(a) C language has been developed by

www.indianfun.org

(1) Ken Thompson
(2) Dennis Ritchie
(3) Peter Norton
(4) Martin Richards

Answer:

(2) Dennis Ritchie

(b) C language has been developed at

(1) Microsoft Corp., USA
(2) AT & T Bell Labs, USA
(3) Borland International, USA
(4) IBM, USA

Answer:

(2) AT & T Bell Labs, USA

(c) C language came into existence in the year

(1) 1971
(2) 1957
(3) 1972
(4) 1983

Answer:

(3) 1972

(d) C is a

(1) Middle level language
(2) High level language
(3) Low level language
(4) None of the above

Answer:

(1) Middle level language

(e) C can be used on

www.indianfun.org

(1) Only MS-DOS operating system
(2) Only Unix operating system
(3) Only Xenix operating system
(4) All the above

Answer:

(4) All the above

(f) C programs are converted into machine language with the help of

(1) An interpreter
(2) A compiler
(3) An operating system
(4) None of the above

Answer:

(2) A compiler

(g) The real constant in C can be expressed in which of the following forms

(1) Fractional form only
(2) Exponential form only
(3) ASCII form only
(4) Both fractional and exponential forms

Answer:

(4) Both fractional and exponential forms

(h) A character variable can at a time store

(1) 1 character
(2) 8 characters
(3) 254 characters
(4) None of the above

Answer:

(1) 1 character

(i) Which of the following is NOT a character constant

(1) 'Thank You'
(2) 'Enter values of P, N, R'
(3) '23.56E-03'
(4) All the above

Answer:

(4) All the above

(j) The maximum value that an integer constant can have is

(1) -32767
(2) 32767
(3) 1.7014e+38
(4) -1.7014e+38

Answer:

(2) 32767

(k) The maximum width of a C variable name can be

(1) 6 characters
(2) 8 characters
(3) 10 characters
(4) 20 characters

Answer:

(2) 8 characters. New C compilers permit upto 32 characters

(l) A C variable cannot start with

(1) An alphabet
(2) A number
(3) A special symbol
(4) Both (2) & (3) above

Answer:

(4) Both (2) & (3) above

(m) Which of the following statement is wrong

(1) mes = 123.56 ;
(2) con = 'T' * 'A' ;
(3) this = 'T' * 20 ;
(4) 3 + a = b ;

Answer:

(4) 3 + a = b ;

(n) Which of the following shows the correct hierarchy of arithmetic operations in C

(1) ( ), **, * or /, + or -
(2) ( ), **, *, /, +, -
(3) ( ), **, /, *, +, -
(4) ( ), / or *, - or +

Answer:

(4) ( ), / or *, - or +

(o) In b = 6.6 / a + (2 * a + (3 * c) / a * d) / (2 / n) ; which operation will be performed first?

(1) 6.6 / a
(2) 2 * a
(3) 3 * c
(4) 2 / n

Answer:

(3) 3 * c

(p) Which of the following is allowed in a C Arithmetic instruction

(1) [ ]
(2) { }

www.indianfun.org

(3) ( )
(4) None of the above

Answer:

(3) ( )

(q) Which of the following statements is false

(1) Each new C instruction has to be written on a separate line
(2) Usually all C statements are entered in small case letters
(3) Blank spaces may be inserted between two words in a C statement
(4) Blank spaces cannot be inserted within a integer variable

Answer:

(1) Each new C instruction has to be written on a separate line

(r) If **a** is an integer variable, a = 5 / 2 ; will return a value

(1) 2.5
(2) 3
(3) 2
(4) 0

Answer:

(2) 2

(s) The expression, a = 7 / 22 * (3.14 + 2) * 3 / 5 ; evaluates to

(1) 8.28
(2) 6.28
(3) 3.14
(4) 0

Answer:

(4) 0

www.indianfun.org

(t) The expression, a = 30 * 1000 + 2768 ; evaluates to

(1) 32768
(2) -32768
(3) 113040
(4) 0

Answer:

(2) -32768

(u) The expression x = 4 + 2 % -8 evaluates to

(1) -6
(2) 6
(3) 4
(4) None of the above

Answer:

(2) 6

(v) Hierarchy decides which operator

(1) is most important
(2) is used first
(3) is fastest
(4) operates on largest numbers

Answer:

(2) is used first

[G] Write C programs for the following:

(a) Ramesh's basic salary is input through the keyboard. His dearness allowance is 40% of basic salary, and house rent allowance is 20% of basic salary. Write a program to calculate his gross salary.

Program:


```
/* To calculate the gross salary of Ramesh */

#include <stdio.h>
#include <conio.h>

main( )
{
    float bp, da, hra, grpay ;

    clrscr( ) ; /* Clears Screen */
    printf ( "\nEnter the Basic Pay of Ramesh : " ) ;
    scanf ( "%f", &bp ) ;

    da = 0.4 * bp ;
    hra = 0.2 * bp ;
    grpay = bp + da + hra ; /* Gross Pay = sum of basic &
                               all allowances */

    printf ( "\nBasic Pay of Ramesh = %f", bp ) ;
    printf ( "\nDearness Allowance = %f", da ) ;
    printf ( "\nHouse Rent Allowance = %f", hra ) ;
    printf ( "\nGross Pay of Ramesh is %f", grpay ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ; /* reads a character from keyboard */
}
```

(a) The distance between two cities (in km.) is input through the keyboard. Write a program to convert and print this distance in meters, feet, inches and centimeters.

Program:

```
/* Conversion of distance */
```

```
#include <stdio.h>
#include <conio.h>

main( )
{
    float  km, m , cm, ft, inch ;

    clrscr( ) ;
    printf ( "\nEnter the distance in Kilometers : " ) ;
    scanf ( "%f", &km ) ;

    m = km * 1000 ;
    cm = m * 100 ;
    inch = cm / 2.54 ;
    ft = inch / 12 ;

    printf ( "\nDistance in meters = %f", m ) ;
    printf ( "\nDistance in centimeter = %f", cm ) ;
    printf ( "\nDistance in feet = %f", ft ) ;
    printf ( "\nDistance in inches = %f", inch ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(b) If the marks obtained by a student in five different subjects are input through the keyboard, find out the aggregate marks and percentage marks obtained by the student. Assume that the maximum marks that can be obtained by a student in each subject is 100.

Program:

```
/* Calculation of aggregate & percentage marks */

#include <stdio.h>
```



```
#include <conio.h>

main( )
{
    int  m1, m2, m3, m4, m5, aggr ;
    float  per ;

    clrscr( ) ;
    printf ( "\nEnter marks in 5 subjects :" ) ;
    scanf ( "%d %d %d %d %d", &m1, &m2, &m3, &m4, &m5 ) ;

    aggr = m1 + m2 + m3 + m4 + m5 ;
    per = aggr / 5 ;

    printf ( "\nAggregate Marks = %d ", aggr ) ;
    printf ( "\nPercentage Marks = %f", per ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(c) Temperature of a city in fahrenheit degrees is input through the keyboard. Write a program to convert this temperature into centigrade degrees.

Program:

```
/* Conversion of temperature from Fahrenheit to Centigrade */

#include <stdio.h>
#include <conio.h>

main( )
{
    float  fr, cent ;
```

```
    clrscr( ) ;
    printf ( "\nEnter the temperature (F) : " ) ;
    scanf ( "%f", &fr ) ;

    cent = 5.0 / 9.0 * ( fr - 32 ) ;
    printf ( "\nTemperature in centigrade = %f", cent ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) The length & breadth of a rectangle and radius of a circle are input through the keyboard. Write a program to calculate the area & perimeter of the rectangle, and the area & circumference of the circle.

Program:

```
/* Calculation of perimeter & area of rectangle and circle */

#include <stdio.h>
#include <conio.h>

main( )
{
    int l, b, r, area1, perimeter ;
    float area2, circum ;

    clrscr( ) ;
    printf ( "\nEnter Length & Breadth of Rectangle " ) ;
    scanf ( "%d %d", &l, &b ) ;

    area1 = l * b ; /* Area of a rectangle */
    perimeter = 2 * l + 2 * b ; /* Perimeter of a rectangle */

    printf ( "\nArea of Rectangle = %d ", area1 ) ;
```


www.indianfun.org


```
    printf ( "\nPerimeter of Rectangle = %d ", perimeter) ;

    printf ( "\n\nEnter Radius of circle " ) ;
    scanf ( "%d", &r ) ;

    area2 = 3.14 * r * r ; /* Area of Circle */
    circum = 2 * 3.14 * r ; /* Circumference of a circle */

    printf ( "Area of Circle = %f ", area2 ) ;
    printf ( "\nCircumference of Circle = %f", circum ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(e) Two numbers are input through the keyboard into two locations C and D. Write a program to interchange the contents of C and D.

Program:

```
/* Interchanging of contents of two variables c & d */

#include <stdio.h>
#include <conio.h>

main( )
{
    int c, d, e ;

    clrscr( ) ;
    printf ( "\nEnter the number at location C: " ) ;
    scanf ( "%d", &c ) ;

    printf ( "\nEnter the number at location D: " ) ;
    scanf ( "%d", &d ) ;
```

```
    /* Interchange the contents of two variables using a third variable as
       temporary store */
    e = c ;
    c = d ;
    d = e ;

    printf ( "\nNew Number at location C = %d", c ) ;
    printf ( "\nNew Number at location D = %d", d ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(f) If a five-digit number is input through the keyboard, write a program to calculate the sum of its digits.

(Hint: Use the modulus operator '%')

The division operator /, returns the quotient on dividing one number by another, whereas the modulus operator % gives the remainder on dividing one number by another. For example,

a = 12 % 3 would store 0 in **a**

a = 13 % 5 would store 3 in **a**

Program:

```
/* sum of digits of a 5 digit number */

#include <stdio.h>
#include <conio.h>

main( )
{
```



```
    int num, a, n ;
    int sum = 0 ; /*sum initialised to zero as otherwise it will contain a
                    garbage value*/

    clrscr( ) ;
    printf ( "\nEnter a 5 digit number(less than 32767)" ) ;
    scanf ( "%d", &num ) ;

    a = num % 10 ; /* last digit extracted as remainder */
    n = num /10 ; /* Remaining digits */
    sum = sum + a ; /* sum updated with addition of extracted digit */

    a = n % 10 ; /* 4 th digit */
    n = n /10 ;
    sum = sum + a ;

    a = n % 10 ; /* 3 rd digit */
    n = n /10 ;
    sum = sum + a ;

    a = n % 10 ; /* 2 nd digit */
    n = n /10 ;
    sum = sum + a ;

    a = n % 10 ; /* 1 st digit */
    sum = sum + a ;

    printf ( "\nThe sum of the 5 digits of %d is %d", num, sum ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(g) If a five-digit number is input through the keyboard, write a program to reverse the number.

Program:

```
/* To reverse the digits of a 5-digit number */

#include <stdio.h>
#include <conio.h>

main( )
{
    int n, a, b ;
    long int  revnum = 0 ;
    /* Initialised otherwise it will contain garbage value.Declared as long
       as after reversing it may not be in the int range */

    clrscr( ) ;
    printf ( "\nEnter a five digit number ( less than 32767 ) " ) ;
    scanf ( "%d", &n ) ;

    a = n % 10 ;  /* last digit */
    n = n / 10 ;  /* Remaining digits */
    revnum = revnum + a * 10000L ;  /* revnum updated with
                                       value of extracted digit */

    a = n % 10 ;  /* 4 th digit */
    n = n / 10 ;  /* Remaining digits */
    revnum = revnum + a * 1000 ;

    a = n % 10 ;  /* 3 rd digit */
    n = n / 10 ;  /* Remaining digits */
    revnum = revnum + a * 100 ;

    a = n % 10 ;  /* 2 nd digit */
    n = n / 10 ;  /* Remaining digits */
    revnum = revnum + a * 10 ;

    a = n % 10 ;  /* 1 st digit */

    revnum = revnum + a ;
```



```
    printf ( "\nThe reversed number is %ld", revnum ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(h) If a four-digit number is input through the keyboard, write a program to obtain the sum of the first and last digit of this number.

Program:

```
/* To sum the 1st & last digit of a four digit number */

#include <stdio.h>
#include <conio.h>

main( )
{
    int n, a, sum = 0 ;

    clrscr( ) ;
    printf ( "\nEnter a four digit number " ) ;
    scanf ( "%d", &n ) ;

    a = n / 1000 ;  /* 1 st digit */
    sum = sum + a ;  /* sum updated with addition of 1 st digit */

    a = n % 10 ;  /* Last digit */
    sum = sum + a ;  /* sum updated with addition of last digit */

    printf ( "\nSum of first and last digit of %d = %d",n, sum ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

Chapter 2
The Decision Control Structure

if, *if-else*, Nested *if-elses*

[A] What will be the output of the following programs?

(a)
```
main( )
{
    int  a = 300, b, c ;
    if ( a >= 400 )
        b = 300 ;
        c = 200 ;
        printf ( "\n%d %d", b, c ) ;
}
```

Output:

b will contain some garbage value and c will be equal to 200

(b)
```
main( )
{
    int  a = 500, b, c ;
    if ( a >= 400 )
        b = 300 ;
        c = 200 ;
        printf ( "\n%d %d", b, c ) ;
```

```
}
```

Output:

300 200

(c)
```
main( )
{
    int x = 10, y = 20 ;
    if ( x == y ) ;
        printf ( "\n%d %d", x, y ) ;
}
```

Output:

10 20

(d)
```
main( )
{
    int x = 3, y = 5 ;
    if ( x == 3 )
        printf ( "\n%d", x ) ;
    else ;
        printf ( "\n%d", y ) ;
}
```

Output:

3

(e)
```
main( )
{
    int x = 3 ;
    float y = 3.0 ;
    if ( x == y )
        printf ( "\nx and y are equal" ) ;
    else
```



```
        printf ( "\nx and y are not equal" ) ;
}
```

Output:

x and y are equal

(f)
```
main( )
{
    int x = 3, y, z ;
    y = x = 10 ;
    z = x < 10 ;
    printf ( "\nx = %d y = %d z = %d", x, y, z ) ;
}
```

Output:

10 10 0

(g)
```
main( )
{
    int k = 35 ;
    printf ( "\n%d %d %d", k == 35, k = 50, k > 40 ) ;
}
```

Output:

0 50 0

[B] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    float a = 12.25, b = 12.52 ;
    if ( a = b )
        printf ( "\na and b are equal" ) ;
}
```

Error. For comparison use == and not =.

(b)
```
main( )
{
    int j = 10, k = 12 ;
    if ( k >= j )
    {
        {
            k = j ;
            j = k ;
        }
    }
}
```

No Error. Any number of pairs of braces can be used.

(c)
```
main( )
{
    if ( 'X' < 'x' )
        printf ( "\nascii value of X is smaller than that of x" ) ;
}
```

No Error. Ascii values of x and X are being compared.

(d)
```
main( )
{
    int x = 10 ;
    if ( x >= 2 ) then
        printf ( "\n%d", x ) ;
}
```

Error. 'then' cannot be used in C.

(e)
```
main( )
{
    int x = 10 ;
    if x >= 2
        printf ( "\n%d", x ) ;
}
```

Error. The condition after if should always be in parenthesis.

(f)
```
main( )
{
    int x = 10, y = 15 ;
    if ( x % 2 = y % 3 )
        printf ( "\nCarpathians" ) ;
}
```

Error. For comparison use == and not =.

[C] Attempt the following:

(a) If cost price and selling price of an item is input through the keyboard, write a program to determine whether the seller has made profit or incurred loss. Also determine how much profit he made or loss he incurred.

Program:

```
/* Calculate profit or loss */

#include <stdio.h>
#include <conio.h>

main( )
{
    float cp, sp, p, l ;

    clrscr( ) ;
    printf ( "Enter cost price and selling price : " ) ;
    scanf ( "%f %f", &cp, &sp ) ;

    p = sp - cp ;  /* Profit = Selling Price - Cost Price */
```

```
    l = cp - sp ; /* Loss =  Cost Price - Selling Price */

    if ( p > 0 )
        printf ( "The seller has made a profit of Rs.%f", p ) ;

    if ( l > 0 )
        printf ( "The seller is in loss by Rs.%f", l ) ;

    if ( p == 0 )
        printf ( "There is no loss, no profit" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(b) Any integer is input through the keyboard. Write a program to find out whether it is an odd number or even number.

Program:

```
/* check whether a number is even or odd */

#include <stdio.h>
#include <conio.h>

main( )
{
   int n ;

   clrscr( ) ;
   printf ( "Enter any number" ) ;
   scanf ( "%d", &n ) ;

   if ( n % 2 == 0 ) /* Remainder after division by 2 */
       printf ( "\nThe number is even" ) ;
   else
```



```
       printf ( "\nThe number is odd" ) ;

   printf ( "\n\n\n\n\nPress any key to exit..." ) ;
   getch( ) ;
}
```

(c) Any year is input through the keyboard. Write a program to determine whether the year is a leap year or not.

(Hint: Use the % (modulus) operator)

Program:

```
/* To check whether the year is leap or not */
/* The year is Leap if it is fully divisible by 100 & 400 */
/* Otherwise, */
/* The year is Leap if it is fully divisible by 4 */

#include <stdio.h>
#include <conio.h>

main( )
{
   int yr ;

   clrscr( ) ;

   printf ( "Enter a year:" ) ;
   scanf ( "%d", &yr ) ;

   if ( yr % 100 == 0 )
   {
      if ( yr % 400 == 0 )
         printf ( "\nLeap year" ) ;
      else
         printf ( "\nNot a Leap year" ) ;
```

```
    }
    else
    {
        if ( yr % 4 == 0 )
            printf ( "\nLeap year" ) ;
        else
            printf ( "\nNot a leap year" ) ;
    }

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) According to the Gregorian calendar, it was Monday on the date 01/01/1900. If any year is input through the keyboard write a program to find out what is the day on 1st January of this year.

Program:

```
/* To calculate the day on 1st Jan of any year */

#include <stdio.h>
#include <conio.h>

main( )
{
    int leapdays, firstday, yr ;
    long int normaldays, totaldays ;

    clrscr( ) ;
    printf ( "Enter year:" ) ;
    scanf ( "%ld", &yr ) ;

    normaldays = ( yr - 1 ) * 365L ;
    leapdays = ( yr - 1 ) / 4 - ( yr - 1 ) / 100 + ( yr - 1 ) / 400 ;
```



```
    totaldays = normaldays + leapdays ;
    firstday = totaldays % 7 ;

    if ( firstday == 0 )
        printf ( "\nMonday" ) ;

    if ( firstday == 1 )
        printf ( "\nTuesday" ) ;

    if ( firstday == 2 )
        printf ( "\nWednesday" ) ;

    if ( firstday == 3 )
        printf ( "\nThursday" ) ;

    if ( firstday == 4 )
        printf ( "\nFriday" ) ;

    if ( firstday == 5 )
        printf ( "\nSaturday" ) ;

    if ( firstday == 6 )
        printf ( "\nSunday" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(e) A five-digit number is entered through the keyboard. Write a program to obtain the reversed number and to determine whether the original and reversed numbers are equal or not.

Program:

```
/* To check whether a number and its reversed number are equal */
```

```
#include <stdio.h>
#include <conio.h>

main( )
{
    int n, a, b, num;
    long int revnum = 0 ;
    /* Initialised otherwise it will contain garbage value.*/
    /* Declared as long since after reversing, it may not be in the int
       range */

    clrscr( ) ;
    printf ( "\nEnter a five digit number ( less than 32767 )" ) ;
    scanf ( "%d", &n ) ;

    num = n ; /* Entered number stored for comparison later */
    a = n % 10 ; /* last digit */
    n = n / 10 ; /* Remaining digits */
    revnum = revnum + a * 10000L; /* revnum updated with
                                     value of extracted digit */

    a = n % 10 ; /* 4 th digit */
    n = n / 10 ; /* Remaining digits */
    revnum = revnum + a * 1000;

    a = n % 10 ; /* 3 rd digit */
    n = n / 10 ; /* Remaining digits */
    revnum = revnum + a * 100;

    a = n % 10 ; /* 2 nd digit */
    n = n / 10 ; /* Remaining digits */
    revnum = revnum + a * 10 ;

    a = n % 10 ; /* 1 st digit */

    revnum = revnum + a ;

    if ( revnum == num )
```



```
        printf ( "\nGiven number and its reversed number are equal" );
    else
        printf ( "\nGiven number and its reversed number are not equal" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

Logical Operators

[D] if x = 11, y = 6, z = 1, find the values of the expressions in the following table:

Expression	Value
x > 9 && y != 3	1
x == 5 || y != 3	1 (Answer)
! (x > 14)	1 (Answer)
! (x > 9 && y != 23)	0 (Answer)
5 && y != 8 || 0	1 (Answer)

[E] What will be the output of the following programs:

(a)
```
main( )
{
    int i = 4, z = 12 ;
    if ( i = 5 || z > 50 )
        printf ( "\nDean of students affairs" ) ;
    else
        printf ( "\nDosa" ) ;
}
```

Output:

Dean of students affairs

(b) main( )

```
{
    int i = 4, z = 12 ;
    if ( i = 5 && z > 5 )
        printf ( "\nLet us C" ) ;
    else
        printf ( "\nWish C was free !" ) ;
}
```

Output:

Let us C

(c) main( )

```
{
    int i = 4, j = -1, k = 0, w, x, y, z ;
    w = i || j || k ;
    x = i && j && k ;
    y = i || j && k ;
    z = i && j || k ;
    printf ( "\nw = %d x = %d y = %d z = %d", w, x, y, z ) ;
}
```

Output:

1 0 1 1

(d) main( )

```
{
    int i = 4, j = -1, k = 0, y, z ;
    y = i + 5 && j + 1 || k + 2 ;
    z = i + 5 || j + 1 && k + 2 ;
    printf ( "\ny = %d z = %d", y, z ) ;
}
```

Output:

1 1

(e) main( )

```
{
    int i = -3, j = 3 ;
    if ( !i + !j * 1 )
        printf ( "\nMassaro" ) ;
    else
        printf ( "\nBennarivo" ) ;
}
```

Output:

Bennarivo

[F] Point out the errors, if any, in the following programs:

(a) /* This program is an example of using Logical operators */

```
main( )
{
    int i = 2, j = 5 ;
    if ( i == 2 && j == 5 )
        printf ( "\nSatisfied at last" ) ;
}
```

No Error

(b) main( )

```
{
    int code, flag ;
    if ( code == 1 & flag == 0 )
        printf ( "\nThe eagle has landed" ) ;
}
```

Error. To combine two conditions use && and not &.

(c)
```
main( )
{
    char spy = 'a', password = 'z' ;
    if ( spy == 'a' or password == 'z' )
        printf ( "\nAll the birds are safe in the nest" ) ;
}
```

Error. 'or' cannot be used to combine conditions . Use ||.

(d)
```
main( )
{
    int i = 10, j = 20 ;
    if ( i = 5 ) && if ( j = 10 )
        printf ( "\nHave a nice day" ) ;
}
```

Error. Condition should be if (i == 5 && j == 10).

[G] Attempt the following:

(a) Any year is entered through the keyboard, write a program to determine whether the year is leap or not. Use the logical operators **&&** and ||.

Program:

```
/* Determine leap or not a leap year */

#include <stdio.h>
#include <conio.h>

main( )
{
    int year ;
    clrscr( ) ;
    printf ( "\nEnter year:" ) ;
    scanf ( "%d", &year ) ;

    if ( year % 400 == 0 || year % 100 != 0 && year % 4 == 0 )
        printf ( "\nLeap year" ) ;
    else
        printf ( "\nNot a leap year" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```


(b) Any character is entered through the keyboard, write a program to determine whether the character entered is a capital letter, a small case letter, a digit or a special symbol.

The following table shows the range of ascii values for various characters.

Characters	Ascii Values
A – Z	65 – 90
a – z	97 – 122
0 – 9	48 – 57
special symbols	0 - 47, 58 - 64, 91 - 96, 123 - 127

Program:

```
/* To check type of character entered from keyboard */

#include <stdio.h>
#include <conio.h>

main( )
```

```
{
    char ch ;

    clrscr( ) ;
    printf ( "\nEnter a character from the keyboard" ) ;
    scanf ( "%c", &ch ) ;

    if ( ch >= 65 && ch <= 90 )
        printf ( "\nThe character is an uppercase letter" ) ;

    if ( ch >= 97 && ch <= 122 )
        printf ( "\nThe character is a lowercase letter" ) ;

    if ( ch >= 48 && ch <= 57 )
        printf ( "\nThe character is a digit" ) ;

    if ( ( ch >= 0 && ch < 48 ) || ( ch > 57 && ch < 65 )
        || ( ch > 90 && ch < 97 ) || ch > 122 )
        printf ( "\nThe character is a special symbol" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(c) An Insurance company follows following rules to calculate premium.

(1) If a person's health is excellent and the person is between 25 and 35 years of age and lives in a city and is a male then the premium is Rs. 4 per thousand and his policy amount cannot exceed Rs. 2 lakhs.
(2) If a person satisfies all the above conditions except that the sex is female then the premium is Rs. 3 per thousand and her policy amount cannot exceed Rs. 1 lakh.
(3) If a person's health is poor and the person is between 25 and 35 years of age and lives in a village and is a male

then the premium is Rs. 6 per thousand and his policy cannot exceed Rs. 10,000.
(4) In all other cases the person is not insured.

Write a program to Output whether the person should be insured or not, his/her premium rate and maximum amount for which he/she can be insured.

Program:

```
/* Check whether a person is insured or not */

#include <stdio.h>
#include <conio.h>

main( )
{
    int age, pr, policy ;
    char health, resident, sex ;

    clrscr( ) ;

    printf ( "\n\nEnter the values of :\n\n" ) ;
    printf ( "sex (M/F), health excellent/poor (E/P)\n" ) ;
    printf ( "Resident of city/village (C/V) and age:" ) ;
    scanf ( "%c %c %c %d", &sex, &health, &resident, &age ) ;

    if ( ( sex == 'M'|| sex == 'm') && ( health == 'E' || health == 'e' )
     && ( resident == 'C'|| resident == 'c') && age >= 25 && age <=
     35 )

    /* These conditions will work even if lowercase letters are entered */
    {
        printf ( "\nThe person is insured" ) ;
        printf ( "\nPremium = Rs.4 per thousand" ) ;
        printf ( "\nMax insurance.amount = Rs.2 lacs" ) ;
```

```
            printf ( "\n\n\n\n\nPress any key to exit..." ) ;
            getch( ) ;

            exit( ) ;  /* Terminates Program execution */
        }

        if ( ( sex == 'F'|| sex == 'f' ) && ( health == 'E' || health == 'e' )
          && ( resident == 'C' || resident == 'c' )&&age >= 25 && age <=
          35 )
        {
            printf ( "\nThe person is insured" ) ;
            printf ( "\nPremium = Rs.3 per thousand" ) ;
            printf ( "\nMax insurance amount = Rs.1 lac" ) ;

            printf ( "\n\n\n\n\nPress any key to exit..." ) ;
            getch( ) ;

            exit( ) ;
        }

        if ( ( sex == 'M'|| sex == 'm' ) && ( health == 'P' || health == 'p' )
          && ( resident == 'V' || resident == 'v' )&&age >= 25 && age <=
          35 )
        {
            printf ( "\nThe person is insured" ) ;
            printf ( "\nPremium = Rs.6 per thousand" ) ;
            printf ( "\nMax insurance amount = Rs.10100" ) ;

            printf ( "\n\n\n\n\nPress any key to exit..." ) ;
            getch( ) ;

            exit( ) ;
        }

        printf ( "\nThe person is not insured" ) ;

        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
```



```
    }
```

(d) A certain grade of steel is graded according to the following conditions:

(1) Hardness must be greater than 50
(2) Carbon content must be less than 0.7
(3) Tensile strength must be greater than 5600

The grades are as follows:

Grade is 10 if all three conditions are met
Grade is 9 if conditions (i) and (ii) are met
Grade is 8 if conditions (ii) and (iii) are met
Grade is 7 if conditions (i) and (iii) are met
Grade is 6 if only one condition is met
Grade is 5 if none of the conditions are met

Write a program, which will require the user to give values of hardness, carbon content and tensile strength of the steel under consideration and Output the grade of the steel.

Program:

```
/* To check the grade of steel */

#include <stdio.h>
#include <conio.h>

main( )
{
    float hard, carbon, tensile ;

    clrscr( ) ;
    printf ( "Hardness of steel" ) ;
    scanf ( "%f", &hard ) ;
```

```
printf ( "Carbon content" ) ;
scanf ( "%f", &carbon ) ;

printf ( "Tensil strength" ) ;
scanf ( "%f", &tensile ) ;

if ( hard > 50 && carbon < 0.7 && tensile > 5600 )
{
    printf ( "\nGrade 10" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

    exit( ) ;  /* Terminates the execution */
}

if ( hard > 50 && carbon < 0.7 && tensile < 5600 )
{
    printf ( "\nGrade 9" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

    exit( ) ;
}

if ( hard <= 50 && carbon < 0.7 && tensile > 5600 )
{
    printf ( "\nGrade 8" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

    exit( ) ;
}

if ( hard > 50 && carbon >= 0.7 && tensile > 5600 )
{
```



```
    printf ( "\nGrade 7" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

    exit( ) ;
}

if ( hard > 50 || carbon < 0.7 || tensile > 5600 )
{
    printf ( "\nGrade 6" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

    exit( ) ;
}

printf ( "\nGrade 5" ) ;

printf ( "\n\n\n\n\nPress any key to exit..." ) ;
getch( ) ;
}
```

Conditional Operators

[H] What will be the output of the following programs:

(a)
```
main( )
{
    int i = -4, j, num ;
    j = ( num < 0 ? 0 : num * num ) ;
    printf ( "\n%d", j ) ;
}
```

Output:

Unpredictable :: num not initialised

(b)
```
main( )
{
    int k, num = 30 ;
    k = ( num > 5 ? ( num <= 10 ? 100 : 200 ) : 500 ) ;
    printf ( "\n%d", num ) ;
}
```

Output:

30

[I] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    int tag = 0, code = 1 ;
    if ( tag == 0 )
        ( code > 1 ? printf ( "\nHello" ) ? printf ( "\nHi" ) ) ;
    else
        printf ( "\nHello Hi !!" ) ;
}
```

Error. missing : (colon).

(b)
```
main( )
{
    int ji = 65 ;
    printf ( "\nji >= 65 ? %d : %c", ji ) ;
}
```

No Error.

(c)
```
main( )
{
    int i = 10, j ;
    i >= 5 ? ( j = 10 ) : ( j = 15 ) ;
    printf ( "\n%d %d", i, j ) ;
}
```


No Error.

[J] Rewrite the following programs using conditional operators.

(a)
```
main( )
{
    int x, min, max ;
    scanf ( "\n%d %d", &max, &x ) ;
    if ( x > max )
        max = x ;
    else
        min = x ;
}
```

Program:

```
main( )
{
    int x, min, max ;
    scanf ( "%d%d", &max, &x ) ;
    ( x > max ? ( max = x ) : ( min = x ) ) ;
}
```

(b)
```
main( )
{
    int code ;
    scanf ( "%d", &code ) ;
    if ( code > 1 )
        printf ( "\nJerusalem" ) ;
    else
        if ( code < 1 )
            printf ( "\nEddie" ) ;
        else
            printf ( "\nC Brain" ) ;
}
```

Program:

```
main( )
{
```


www.indianfun.org

```
    int code ;
    scanf ( "%d", &code ) ;
    ( code > 1 ? printf ( " Jerusalem " ) : ( code < 1 ?
      printf ( "Eddie" ) : printf ( "C brain " ) ) )
}
```

[K] Attempt the following:

(a) Using conditional operators determine:

(1) Whether the character entered through the keyboard is a lower case alphabet or not.

Program:

```
/* Program to determine whether the character entered is a lower case
        using conditional operators */

#include <stdio.h>
#include <conio.h>

main( )
{
   char ch ;

   clrscr( ) ;

   printf ( "Enter character" ) ;
   scanf ( "%c", &ch ) ;
   ch >= 97 && ch <= 122 ? printf ( "Character entered is lower case" )
   : printf ( "Character entered is not lower case" ) ;

   printf ( "\n\n\n\n\nPress any key to exit..." ) ;
   getch( ) ;
}
```


(2) Whether a character entered through the keyboard is a special symbol or not.

Program:

```
/* Program to determine whether a character entered is a special
        symbol */

#include <stdio.h>
#include <conio.h>

main( )
{
   char ch ;

   clrscr( ) ;
   printf ( "Enter character" ) ;
   scanf ( "%c", &ch ) ;
   ch = ( ( ch >= 0 && ch <= 47 ) || ( ch >= 58 && ch <= 64 ) ||
          ( ch >= 91 && ch <= 96 ) || ( ch >= 123 ) ) ?
          printf ( "Character entered is a special symbol" ) :
          printf ( "Character entered is not a special symbol" ) ;

   printf ( "\n\n\n\n\nPress any key to exit..." ) ;
   getch( ) ;
}
```

(b) Write a program using conditional operators to determine whether a year entered through the keyboard is a leap year or not.

Program:

```
/* Program to determine whether a year is leap year or not using
        conditinal operators*/

#include <stdio.h>
#include <conio.h>
```

```
main( )
{
    int year ;

    clrscr( ) ;
    printf ( "Enter Year" ) ;
    scanf ( "%d" , &year ) ;
    year % 100 == 0 ? ( year % 400 == 0 ? printf ( " Leap Year " )
    : printf ( "Not a Leap Year" ) ) :( year % 4 == 0 ?
    printf ( "Leap Year" ) : printf ( "Not A Leap Year" ) ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

Chapter 3

The Loop Control Structure

while Loop

[A] What will be the output of the following programs:

(a)
```
main( )
{
    int j ;

    while ( j <= 10 )
    {
        printf ( "\n%d", j ) ;
        j = j + 1 ;
    }
}
```

Output:

Output cannot be predicted since j has not been initialised.

(b)
```
main( )
{
    int i = 1 ;

    while ( i <= 10 ) ;
    {
        printf ( "\n%d", i ) ;
        i++ ;
```

```
    }
}
```

Output:

No Output Indefinite while loop because of ; at the end of while

(c)
```
main( )
{
    int j ;

    while ( j <= 10 )
    {
        printf ( "\n%d", j ) ;
        j = j + 1 ;
    }
}
```

Output:

Output cannot be predicted since j has not been initialised

(d)
```
main( )
{
    int x = 1 ;

    while ( x == 1 )
    {
        x = x - 1 ;
        printf ( "\n%d", x ) ;
    }
}
```

Output:

0

(e)
```
main( )
{
    int x = 1 ;

    while ( x == 1 )
        x = x - 1 ;
    printf ( "\n%d", x ) ;
}
```

Output:

0

(f)
```
main( )
{
    char x ;

    while ( x = 0 ; x <= 255 ; x++ )
        printf ( "\nAscii value %d Character %c", x, x ) ;
}
```

Output:

Error, while instead of for.

(g)
```
main( )
{
    int x = 4, y, z ;
    y = --x ;
    z = x-- ;
    printf ( "\n%d %d %d", x, y, z ) ;
}
```

Output:

2 3 3

(h)
```
main( )
{
    int x = 4, y = 3, z ;
    z = x-- -y ;
    printf ( "\n%d %d %d", x, y, z ) ;
```

```
}
```

Output:

3 3 1

(i)
```
main( )
{
    while ( 'a' < 'b' )
        printf ( "\nmalyalam is a palindrome" ) ;
}
```

Output:

malyalam is a palindrome will be printed indefinitely

(j)
```
main( )
{
    int i = 10 ;
    while ( i = 20 )
        printf ( "\nA computer buff!" ) ;
}
```

Output:

A computer buff! will be printed indefinitely

[B] Pick the odd one out:

1. a = a + 1 ;
2. a += 1 ;
3. a++ ;
4. a =+ 1 ;

Answer:

4. a =+ 1 ;

[C] Attempt the following:

(a) Write a program to calculate overtime pay of 10 employees. Overtime is paid at the rate of Rs. 12.00 per hour for every hour worked above 40 hours. Assume that employees do not work for fractional part of an hour.

Program:

```
/* Program to find overtime pay of 10 employee.*/

#include <stdio.h>
#include <conio.h>

main( )
{
    float otpay ;
    int hour, i = 1 ;

    clrscr( ) ;

    while ( i <= 10 ) /* Loop for 10 employees */
    {
        printf ( "\nEnter no. of hours worked \n" ) ;
        scanf ( "%d", &hour ) ;

        if ( hour >= 40 )
        {
            otpay = ( hour - 40 ) * 12 ;
            printf ( "\nNo of hours worked = %d \nOvertime
                    pay = Rs.%f", hour, otpay ) ;
        }
        else
        {
            otpay = 0 ;
            printf ( "\nNo of hours worked (%d) is less than
                    40 Hrs.\nHence no overtime pay", hour ) ;
```

```
        }
        i++ ;
    }
    printf ("\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(b) Write a program to find the factorial value of any number entered through the keyboard.

Program:

```
/* Calculation of factorial of any number */

#include <stdio.h>
#include <conio.h>

main( )
{
    int num, i = 1 ;
    unsigned long int fact = 1 ; /* As factorial of a number greater
                                    than 7 is beyond int range */

    clrscr( ) ;
    printf ( "Enter any number ( less than 34 )" ) ;
    /* Factorial of 34 is beyond range */
    scanf ( "%d", &num ) ;

    while ( i <= num )
    {
        fact = fact * i ;
        i++ ;
    }
    printf ( "\nfactorial of %d = %lu", num, fact ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
```



```
    getch( ) ;
}
```

(c) Two numbers are entered through the keyboard. Write a program to find the value of one number raised to the power of another.

Program:

```
/* Program to find power of a number using WHILE loop */

#include <stdio.h>
#include <conio.h>

main( )
{
    int x, y, i = 1 ;
    long int power = 1 ;

    clrscr( ) ;
    printf ( "Enter two numbers:" ) ;
    scanf ( "%d %d", &x, &y ) ;

    while ( i <= y )
    {
        power = power * x ;
        i++ ;
    }
    printf ( "\n%d to the power %d is %ld", x, y, power ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) Write a program to print all the ascii values and their equivalent characters using a while loop. The ascii values vary from 0 to 255.

Program:

```
/* Program to print ascii values and their corresponding characters */

#include <stdio.h>
#include <conio.h>

main( )
{
    int i = 0 ;

    clrscr( ) ;

    while ( i <= 255 )
    {
        printf ( "%d  %c\n", i, i ) ;  /* Use of a correct format specifier */
        i++ ;
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(e) Write a program to print out all Armstrong numbers between 1 and 500. If sum of cubes of each digit of the number is equal to the number itself, then the number is called an Armstrong number. For example, 153 = (1 * 1 * 1) + (5 * 5 * 5) + (3 * 3 * 3)

Program:

```
/* Generate all ARMSTRONG numbers between 1 & 500 */

#include <stdio.h>
#include <conio.h>

main( )
{
    int i = 1, a, b, c ;

    clrscr( ) ;
    printf ( "ARMSTRONG NUMBERS BETWEEN 1 & 500 ARE \n" ) ;

    while ( i <= 500 )
    {
        a = i % 10 ;  /* Extract last digit */
        b = i % 100 ;
        b = ( b - a ) / 10 ;  /* Extract second digit */
        c = i / 100 ;  /* Extract first digit */
        if ( (a * a * a ) + ( b * b * b ) + ( c * c * c ) == i )
            printf ( "%d\n", i ) ;
        i++ ;
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(f) Write a program for a match-stick game between the computer and a user. Your program should ensure that the computer always wins. Rules for the game are as follows:

- There are 21 match-sticks.
- The computer asks the player to pick 1, 2, 3, or 4 match-sticks.
- After the person picks, the computer does its picking.

www.indianfun.org

– Whoever is forced to pick up the last match-stick loses the game.

Program:

```
/* Match stick game */

#include <stdio.h>
#include <conio.h>

main( )
{
   int  m = 21, p, c ;

   clrscr( ) ;

   while ( 1 )
   {
        printf ( "\n\nNo. of matches left = %d", m ) ;
        printf ( "\nPick up 1, 2, 3 or 4 matches" ) ;
        scanf ( "%d", &p ) ;
        if ( p > 4 || p < 1 )
             continue ;

        m = m - p ;
        printf ( "\n\nNo. of matches left = %d", m ) ;

        c = 5 - p ;
        printf ( "\nOut of which computer picked up %d", c ) ;

        m = m - c ;
        if ( m == 1 )
        {
             printf ( "\n\nNumber of matches left %d",m ) ;
             printf ( "\nYou lost the game !!" ) ;
             break ;
        }
```



```
        }
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
   }
```

for, break, continue, do-while

[D] What will be the output of the following programs:

(k)
```
main( )
{
   int  i = 0 ;
   for ( ; i ; )
        printf ( "\nHere is some mail for you" ) ;
}
```

Output:

No Output

(l)
```
main( )
{
   int  i ;
   for ( i = 1 ; i <= 5 ; printf ( "\n%d", i ) ) ;
        i++ ;
}
```

Output:

1 will be printed indefinite number of times.

(m)
```
main( )
{
   int  i = 1, j = 1 ;
   for ( ; ; )
   {
        if ( i > 5 )
             break ;
        else
```

```
            j += i ;
            printf ( "\n%d", j ) ;
            i += j ;
        }
    }
```

Output:

2 5

[E] Answer the following:

(a) The three parts of the loop expression in the for loop are:

the initialisation expression
the testing expression
the incrementation expression

(b) An expression contains relational operators, assignment operators, and arithmetic operators. In the absence of parentheses, they will be evaluated in which of the following order:

(1) assignment, relational, arithmetic
(2) arithmetic, relational, assignment
(3) relational, arithmetic, assignment
(4) assignment, arithmetic, relational

Answer:

(2) arithmetic, relational, assignment

(c) The **break** statement is used to exit from:

(1) an **if** statement
(2) a for loop
(3) a program
(4) the **main( )** function

Answer:

(2) a for loop

(d) A do-while loop is useful when we want that the statements within the loop must be executed:

(1) Only once
(2) Atleast once
(3) More than once
(4) one of the above

Answer:

(2) At least once

[F] Attempt the following:

(a) Write a program to print all prime numbers from 1 to 300. (Hint: Use nested loops, break and continue)

Program:

```
/* Program to generate all prime numbers from 1 to 300 */

#include <stdio.h>
#include <conio.h>

main( )
{
    int i, n = 1 ;

    clrscr( ) ;
    printf ( "\nPrime numbers between 1 & 300 are :\n1\t" ) ;

    while ( n <= 300 ) /* Loop to check numbers upto 300 */
    {
        i = 2 ;
        while ( i < n ) /* Loop starting from 2 to the number */
```

```
        {
            if ( n % i == 0 )
                break ;  /* takes control out of the inner while as soon
                            as the number is fully divisible */
            else
                i++ ;
        }

        if ( i == n )
            printf ( "%d\t", n ) ;  /* tab is used to print all numbers in
                                       one screen */
        n++ ;
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(b) Write a program to fill the entire screen with a smiling face. The smiling face has an ascii value 1.

Program:

```
/* Program to fill entire screen with smiling face */

#include <stdio.h>
#include <conio.h>

main( )
{
    int r , c ;

    clrscr( ) ;

    for ( r = 0 ; r <= 24 ; r++ )  /* Fills rows 0 to 24 */
        for ( c = 0 ; c <= 79 ; c++ )  /* Fills columns 0 to 79 */
            printf ( "%c", 1 ) ;
```



```
    printf ( "Press any key to exit..." ) ;
    getch( ) ;
}
```

(c) Write a program to add first seven terms of the following series using for loop:

$$\frac{1}{1!}+\frac{2}{2!}+\frac{3}{3!}+\ldots\ldots\ldots$$

Program:

```
/* Sum of first seven terms of series */

#include <stdio.h>
#include <conio.h>

main( )
{
    int i = 1, j ;
    float fact, sum = 0.0 ;

    clrscr( ) ;

    while ( i <= 7 )
    {
        fact = 1.0 ;
        for ( j = 1 ; j <= i ; j++ )
            fact = fact * j ;

        sum = sum + i / fact ;
        i++ ;

    }
    printf ( "\nSum of series = %f", sum ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
```

```
    getch( ) ;
}
```

(d) Write a program to produce the following output:

```
A
```

Program:

```
/* Program to produce a typical pattern */

#include <stdio.h>
#include <conio.h>

main( )
{
    int i = 1, x = 71, blanks = 0, j, val , k ;

    clrscr( ) ;

    while ( i <= 7 )
    {
        j = 65 ; /* ASCII value of A */
        val = x ;
        while ( j <= val )
        {
            printf ( "%c", j ) ;
            j++ ;
        }
```



```
        if ( i == 1 )
            val-- ;

        k = 1 ;

        while ( k <= blanks )
        {
            printf ( " " ) ;
            k++ ;
        }
        blanks = 2 * i - 1 ;

        while ( val >= 65 )
        {
            printf ( "%c", val ) ;
            val--;
        }
        printf ( "\n" ) ;
        x-- ;
        i++ ;
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(e) Write a program to generate all combinations of 1 , 2 and 3 using for loop.

Program:

```
/* Program to generate all possible combination of 1 2 3 */

#include <stdio.h>
#include <conio.h>
```


www.indianfun.org

```
main( )
{
    int i = 1, j = 1, k = 1 ;

    clrscr( ) ;

    for ( i = 1 ; i <= 3 ; i++ ) /* 1 st digit */
    {
        for ( j = 1 ; j <= 3 ; j++ ) /* 2 nd digit */
        {
            for ( k = 1 ; k <= 3 ; k++ ) /* 3 rd digit */
                printf ( "%d%d%d\t" , i , j , k ) ;
        }
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(f) According to a study, the approximate level of intelligence of a person can be calculated using the following formula:

i = 2 + (y + 0.5 x)

Write a program, which will produce a table of values of i, y and x, where y varies from 1 to 6, and, for each value of y, x varies from 5.5 to 12.5 in steps of 0.5.

Program:

```
/* Program to produce intelligence table */

#include <stdio.h>
#include <conio.h>

main( )
{
```



```
    int y,n = 0 ; /* n is a line counter */
    float i, x ;

    clrscr( ) ;

    for ( y = 1 ; y <= 6 ; y++ )
    {
        for ( x = 5.5 ; x <= 12.5 ; x += 0.5 )
        {
            i = 2 + ( y + 0.5 * x ) ;
            printf ( "\ny = %d, x = %f i = %f", y, x, i ) ;
            n++ ;

            if ( n >= 24 ) /* Wait after printing 25 lines */
            {
                printf ( "Press any key to continue..." ) ;
                getch( ) ;
                n = 0 ; /* Reset Line counter to 0 */
                clrscr( ) ; /* clear screen */
            }
        }
    }

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

Chapter 4

The Case Control Structure

[A] What will be the output of the following programs:

(a)
```
main( )
{
    char  suite = 3 ;

    switch ( suite )
    {
        case 1 :
            printf ( "\nDiamond" ) ;
        case 2 :
            printf ( "\nSpade" ) ;
        default :
            printf ( "\nHeart") ;
    }
    printf ( "\nI thought one wears a suite" ) ;
}
```

Output:

```
Heart
I thought one wears suite
```

(b) main( )

```
{
    int k, j = 2 ;
    switch ( k = j + 1 )
    {
        case 0 :
            printf ( "\nTailor" ) ;
        case 1 :
            printf ( "\nTutor" ) ;
        case 2 :
            printf ( "\nTramp" ) ;
        default :
            printf ( "\nPure Simple Egghead!" ) ;
    }
}
```

Output:

Pure Simple Egghead

(c) main( )

```
{
    int i = 0 ;

    switch ( i )
    {
        case 0 :
            printf ( "\nTemple is a non-issue" ) ;
        case 1 :
            printf ( "\nAandhi is never stable" ) ;
        case 2 :
            printf ( "\nMandal will ruin India" ) ;
        case 3 :
            printf ( "\nWe want better politicians" ) ;
    }
}
```


Output:

Temple is a non-issue
Aandhi is never stable
Mandal will ruin India
We want better politicians

(d) main( )

```
{
    char ch = 'a' ;

    switch ( ch )
    {
        case 'a' :
        case 'b' :
            printf ( "\nYou entered b" ) ;
        case 'A' :
            printf ( "\na as in ashar" ) ;
    }
}
```

Output:

You entered ba as in ashar

(e) main( )

```
{
    char i = '1' ;

    switch ( i )
    {
        case 0 :
            printf ( "\nFeeding fish" ) ;
        case 1 :
            printf ( "\nWeeding grass" ) ;
        case 2 :
            printf ( "\nMending roof" ) ;
```

```
        default :
            printf ( "\nJust to survive" ) ;
    }
}
```

Output:

Just to survive

[B] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    int suite = 1 ;
    switch ( suite ) ;
    {
        case 0 ;
            printf ( "\nClub" ) ;
        case 1 ;
            printf ( "\nDiamond" ) ;
    }
}
```

Error. Semi-colon after switch statement.

(b)
```
main( )
{
    int temp ;
    scanf ( "%d", &temp ) ;

    switch ( temp )
    {
        case ( temp <= 20 ) :
            printf ( "\nOoooooohhhh! Damn cool!" ) ;
        case ( temp >> 20 && temp <= 30 ) :
            printf ( "\nRain rain here again!" ) ;
        case ( temp >> 30 && temp <= 40 ) :
```



```
            printf ( "\nWish I am at Everest" ) ;
        default :
            printf ( "\nGood old nagpur weather" ) ;
    }
}
```

Error. Relational operators cannot be used in cases.

(c)
```
main( )
{
    float a = 3.5 ;

    switch ( a )
    {
        case 0.5 :
            printf ( "\nThe art of C" ) ;
            break ;
        case 1.5 :
            printf ( "\nThe spirit of C" ) ;
            break ;
        case 2.5 :
            printf ( "\nSee through C" ) ;
            break ;
        case 3.5 :
            printf ( "\nSimply c" ) ;
    }
}
```

Error. Floats cannot be used in cases.

[C] Write a menu driven program which has following options:

1. Factorial of a number.
2. Prime or not
3. Odd or even
4. Exit

Make use of **switch** statement.

The outline of this program is given below:

```
/* A menu driven program */
main( )
{
    int choice ;

    while ( 1 )
    {
        printf ( "\n1. Factorial" ) ;
        printf ( "\n2. Prime" ) ;
        printf ( "\n3. Odd/Even" ) ;
        printf ( "\n4. Exit" ) ;
        printf ( "\nYour choice? " ) ;
        scanf ( "%d", &choice ) ;

        switch ( choice )
        {
            case 1 :
                /* logic for factorial of a number */
                break ;
            case 2 :
                /* logic for deciding prime number */
                break ;
            case 3 :
                /* logic for odd/even */
                break ;
            case 4 :
                exit( ) ;
        }
    }
}
```

Note the following points:

(a) The statement **while (1)** puts the entire logic in an indefinite loop. This is necessary since the menu must keep reappearing on the screen once an item is selected and the appropriate action taken.

(b) **exit( )** is a standard library function. When it gets executed the program execution is immediately terminated. Note that a **break** would not have worked here because it would have taken the control only outside the **switch** and not outside the **while**.

Program:

```
/* Menu driven program */

#include <stdio.h>
#include <conio.h>

main( )
{
    int choice, num, i ;
    unsigned long int fact ;

    clrscr( ) ;

    while ( 1 )
    {
        printf ( "\n\n1. Factorial\n" ) ;
        printf ( "2. Prime\n" ) ;
        printf ( "3. Odd / Even\n" ) ;
        printf ( "4. Exit\n" ) ;

        printf ( "\nYour choice ? " ) ;
        scanf ( "%d", &choice ) ;

        switch ( choice )
        {
```

```
case 1 :
    printf ( "\nEnter number" ) ;
    scanf ( "%d", &num ) ;

    fact = 1 ;
    for ( i = 1 ; i <= num ; i++ )
        fact = fact * i ;
    printf ( "\nFactorial value = %lu", fact ) ;
    break ; /* Takes control out of switch */

case 2 :
    printf ( "\nEnter number" ) ;
    scanf ( "%d", &num ) ;

    for ( i = 2 ; i < num ; i++ )
    {
        if ( num % i == 0 )
        {
            printf ( "\nNot a prime number" ) ;
            break ; /* Takes control out of for loop */
        }
    }
    if ( i == num )
        printf ( "\nPrime number" ) ;
    break ; /* Takes control out of switch */

case 3 :
    printf ( "\nEnter number" ) ;
    scanf ( "%d", &num ) ;

    if ( num % 2 == 0 )
        printf ( "\nEven number" ) ;
    else
        printf ( "\nOdd number" ) ;
    break ; /* Takes control out of switch */

case 4 :
    exit( ) ; /* Terminates program execution */
```



```
        }
    }
}
```

Chapter 5

Functions

Simple functions, Passing values between functions

[A] What will be the output of the following programs:

(a)
```
main( )
{
    printf ( "\nOnly stupids use C?" ) ;
    display( ) ;
}
display( )
{
    printf ( "\nFools too use C!" ) ;
    main( ) ;
}
```

Output:

Both the messages will get printed indefinitely

(b)
```
main( )
{
    printf ( "\nC to it that C survives" ) ;
    main( ) ;
}
```

Output:

The message will get printed indefinitely

(c)
```
main( )
{
    int i = 45, c ;
    c = check ( i ) ;
    printf ( "\n%d", c ) ;
}
check ( int ch )
{
    if ( ch >>= 45 )
        return ( 100 ) ;
    else
        return ( 10 * 10 ) ;
}
```

Output:

100

(d)
```
main( )
{
    int i = 45, c ;
    c = multiply ( i * 1000 ) ;
    printf ( "\n%d", c ) ;
}
check ( int ch )
{
    if ( ch >>= 40000 )
        return ( ch / 10 ) ;
    else
        return ( 10 ) ;
}
```

Output:

function multiply not defined

[B] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    int i = 3, j = 4, k, l ;
    k = addmult ( i, j ) ;
    l = addmult ( i, j ) ;
    printf ( "\n%d %d", k, l ) ;
}
addmult ( int ii, int jj )
{
    int kk, ll ;
    kk = ii + jj ;
    ll = ii * jj ;
    return ( kk, ll ) ;
}
```

Error. A function cannot return 2 values at a time.

(b)
```
main( )
{
    int a ;
    a = message( ) ;
}
message( )
{
    printf ( "\nViruses are written in C" ) ;
    return ;
}
```

No Error. But since no value is being returned there is no need to collect it in variable a.

(c)
```
main( )
{
    float a = 15.5 ;
```


www.indianfun.org

```
    char ch = 'C' ;
    printit ( a, ch ) ;
}
printit ( a, ch )
{
    printf ( "\n%f %c", a, ch ) ;
}
```

Error. a and ch should be declared as float and char respectively in the function printit( ).

(d)
```
main( )
{
    message( ) ;
    message( ) ;
}
message( ) ;
{
    printf ( "\nPraise worthy and C worthy are synonyms" ) ;
}
```

Error. Semicolon shouldn't be present immediately after message( ) in the function definition.

(e)
```
main( )
{
    let_us_c( )
    {
        printf ( "\nC is a Cimple minded language !" ) ;
        printf ( "\nOthers are of course no match !" ) ;
    }
}
```

Error. One function cannot be defined within another function.

[C] Answer the following:

(a) Is this a correctly written function:

```
sqr ( a ) ;
int a ;
{
    return ( a * a ) ;
}
```

Answer:

No Semicolon shouldn't be present immediately after sqr( ) in.

(b) State whether the following statements are True or False:

(1) The variables commonly used in C functions are available to all functions in a program.

Answer: False

(2) To return the control back to the calling function we must use the keyword **return**.

Answer: False

(3) The same variable names can used in different functions, without any conflict.

Answer: True

(4) Every called function must contain a **return** statement.

Anwer: False

(5) A function may contain more than one **return** statements.

Answer: True

(6) Each **return** statement in a function may return a different value.

Answer: True

(7) A function can still be useful even if you don't pass any arguments to it and the function doesn't return any value back.

Answer: True

[D] Answer the following:

(a) Write a function to calculate the factorial value of any integer entered through the keyboard.

Program:

```
/* Calculate factorial value of an integer using a function */

#include <stdio.h>
#include <conio.h>

main( )
{
    int num ;
    long factorial, fact( ) ; /* Will work upto num = 19 only */

    clrscr( ) ;
    printf ( "Enter a number:" ) ;
    scanf ( "%d", &num ) ;

    factorial = fact ( num ) ; /* Functon call */
```



```
    printf ( "\nFactorial of %d = %ld", num, factorial ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

long fact ( int num )
{

    int i ;
    long factorial = 1 ;

    for ( i = 1 ; i <= num ; i++ )
        factorial = factorial * i ;
    return ( factorial ) ;
}
```

(b) Write a function **power (a, b)**, to calculate the value of **a** raised to **b**.

Program:

```
/* Program to calculate power of a value */

#include <stdio.h>
#include <conio.h>

main( )
{
    int x, y ;
    long pow, power( ) ;

    clrscr( ) ;
    printf ( "Enter two numbers:") ;
    scanf ( "%d %d", &x, &y ) ;
```

```
    pow = power ( x , y ) ;  /* Function call */
    printf ( "\n%d to the power %d = %d ", x, y, pow ) ;

    printf ( "\n\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

long power ( int x, int y )
{
    int i ;
    long p = 1 ;

    for ( i = 1 ; i <= y ; i++ )
        p = p * x ;
    return ( p ) ;
}
```

(c) Write a general-purpose function to convert any given year into its roman equivalent. The following table shows the roman equivalents of decimal numbers:

Decimal	Roman	Decimal	Roman
1	i	100	c
5	v	500	d
10	x	1000	m
50	l		

Example:
Roman equivalent of 1988 is mdcccclxxxviii
Roman equivalent of 1525 is mdxxv

Program:

```
/* Convert given year into its roman equivalent */

#include <stdio.h>
#include <conio.h>

main( )
{
    int yr ;

    clrscr( ) ;

    printf ( "Enter year:" ) ;
    scanf ( "%d", &yr ) ;

    yr = romanise ( yr, 1000, 'm' ) ;  /* Series of function calls */
    yr = romanise ( yr, 500, 'd' ) ;
    yr = romanise ( yr, 100, 'c' ) ;
    yr = romanise ( yr, 50, 'l' ) ;
    yr = romanise ( yr, 10, 'x' ) ;
    yr = romanise ( yr, 5, 'v' ) ;
    yr = romanise ( yr, 1, 'i' ) ;

    printf ( "\n\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

romanise ( int y, int k, char ch )
{
    int i, j ;

    if ( y == 9 )
    {
        printf ( "ix" ) ;
        return ( y % 9 ) ;
    }
```

```
    if ( y == 4 )
    {
        printf ( "iv" ) ;
        return ( y % 4 ) ;
    }

    j = y / k ;

    for ( i = 1 ; i <= j ; i++ )
        printf ( "%c", ch ) ;
    return ( y - k * j ) ;
}
```

(d) Any year is entered through the keyboard. Write a function to determine whether the year is a leap year or not.

Program:

```
/* Using a function, find out whether a year is leap year or not */

#include <stdio.h>
#include <conio.h>

main( )
{
    int  year ;

    clrscr( ) ;
    printf ( "Enter year:" ) ;
    scanf ( "%d", &year ) ;

    leapyear ( year ) ;  /* Function call */

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```


www.indianfun.org


```
leapyear ( int year )
{
    if ( year % 4 == 0 && year % 100 != 0 || year % 400 == 0 )
        printf ( "%d is leap year", year ) ;
    else
        printf ( "%d is not a leap year", year ) ;
}
```

(e) A positive integer is entered through the keyboard. Write a function to obtain the prime factors of this number.

For example, prime factors of 24 are 2, 2, 2 and 3, whereas prime factors of 35 are 5 and 7.

Program:

```
/* Obtain prime factors of a number */

#include <stdio.h>
#include <conio.h>

main( )
{
    int  num ;

    clrscr( ) ;
    printf ( "Enter number:" ) ;
    scanf ( "%d", &num ) ;

    prime ( num ) ;  /* Function call */

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

prime ( int num )
{
```

www.indianfun.org

```
    int i = 2 ;
    printf ( "Prime factors of %d are", num ) ;

    while ( num != 1 )
    {
        if ( num % i == 0 )
            printf ( "%d ", i ) ;
        else
        {
            i++ ;
            continue ;
        }
        num = num / i ;
    }
}
```

Function Prototypes, Call by Value/Reference, Pointers

[E] What will be the output of the following programs:

(a)
```
main( )
{
    float area ;
    int radius = 1 ;
    area = circle ( radius ) ;
    printf ( "\n%f", area ) ;
}
circle ( int r )
{
    float a ;
    a = 3.14 * r * r ;
    return ( a ) ;
}
```

Output:


```
3.000000
```

(b)
```
main( )
{
    void slogan( ) ;
    int c = 5 ;
    c = slogan( ) ;
    printf ( "\n%d", c ) ;
}
void slogan( )
{
    printf ( "\nOnly He men use C!" ) ;
}
```

Output:

Error message by compiler

[F] Answer the following:

(a) Write a function which receives a **float** and an **int** from **main( )**, finds the product of these two and returns the product which is printed through **main( )**.

Program:

```
/* Program to find product of float and integer */

#include <stdio.h>
#include <conio.h>

main( )
{
    int x ;
    float y, prod ;
    float product ( int, float ) ; /* Function Prototype */
```

```
    clrscr( ) ;
    printf ( "Enter float value and integer value:" ) ;
    scanf ( "%f %d", &y, &x ) ;

    prod = product ( x, y ) ; /* Function call */
    printf ( "\nProduct of %f and %d = %f", y, x, prod ) ;

    printf ( " \n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

float product ( int x, float y )
{
    float p ;
    p = x * y ;
    return ( p ) ;
}
```

(b) Write a function, which receives 5 integers and returns the sum, average and standard deviation of these numbers. Call this function from **main( )** and print the results in **main( )**.

Program:

```
/* Function which returns sum, average and standard deviation */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main ( )
{
    int sum, avg ;
    double stdev ;

    clrscr( ) ;
```



```
    function( &sum, &avg, &stdev ) ; /* Function call by reference */
    printf ( "\nSum = %d \nAverage = %d \nStandard deviation = %lf",
             sum, avg, stdev ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

function ( int *sum, int *avg, double *stdev )
{
    int n1, n2, n3, n4, n5 ;

    printf ( "\nEnter 5 numbers:" ) ;
    scanf ( "%d%d%d%d%d", &n1, &n2, &n3, &n4, &n5 ) ;

    *sum = n1 + n2 + n3 + n4 + n5 ; /* Calculate sum */
    *avg = *sum / 5 ; /* Calculate average */
    /* Calculate standard deviation */
    *stdev = sqrt( ( pow((n1 - *avg ), 2) + pow(( n2 - *avg ), 2) +
                     pow ( ( n3 - *avg ), 2 ) + pow ( ( n4 - *avg ), 2 ) +
                     pow ( ( n5 - *avg ), 2 ) ) / 4 ) ;
}
```

[G] What will be the output of the following programs:

(a)
```
main( )
{
    int i = 5, j = 2 ;
    junk ( i, j ) ;
    printf ( "\n%d %d", i, j ) ;
}

junk ( int i, int j )
{
    i = i * i ;
    j = j * j ;
}
```


www.indianfun.org

Output:

5 2

(b) main( )

```
{
    int i = 5, j = 2 ;
    junk ( &i, &j ) ;
    printf ( "\n%d %d", i, j ) ;
}

junk ( int *i, int *j )
{
    *i = *i * *i ;
    *j = *j * *j ;
}
```

Output:

25 4

(c) main( )

```
{
    int i = 4, j = 2 ;
    junk ( &i, j ) ;
    printf ( "\n%d %d", i, j ) ;
}

junk ( int *i, int j )
{
    *i = *i * *i ;
    j = j * j ;
}
```

Output:

16 2

(d) main( )

```
{
    float a = 13.5 ;
    float *b, *c ;
    b = &a ; /* suppose address of a is 1006 */
    c = b ;
    printf ( "\n%u %u %u", &a, b, c ) ;
    printf ( "\n%f %f %f %f %f", a, *(&a), *&a, *b, *c ) ;
}
```

Output:

1006 1006 1006
13.500000 13.500000 13.500000 13.500000 13.500000
Note : Instead of 1006 you may get some other number

[H] Point out the errors, if any, in the following programs:

(a) main( )

```
{
    int i = 135, a = 135, k ;
    k = pass ( i, a ) ;
    printf ( "\n%d", k ) ;
}

pass ( int j, int b )
int c ;
{
    c = j + b ;
    return ( c ) ;
}
```

Error. The declaration int c should be made inside the brace.

(b) main( )

```
{
```

```
    int p = 23, f = 24 ;
    jiaayjo ( &p, &f ) ;
    printf ( "\n%d %d", p, f ) ;
}

jiaayjo ( int q, int g )
{
    q = q + q ;
    g = g + g ;
}
```

Error. q and g should be declared as integer pointers.

(c) main( )

```
{
    int k = 35, z ;
    z = check ( k ) ;
    printf ( "\n%d", z ) ;
}

check ( m )
{
    int m ;
    if ( m >> 40 )
        return ( 1 ) ;
    else
        return ( 0 ) ;
}
```

Error. The declaration int m should be before the opening brace.

(d) main( )

```
{
    int i = 35, *z ;
```



```
    z = function ( &i ) ;
    printf ( "\n%d", z ) ;
}

function ( int *m )
{
    return ( m + 2 ) ;
}
```

No Error. Suppose address of m is 2000 then z = 2004.

[I] What will be the output of the following programs:

(a) main( )

```
{
    int i = 0 ;
    i++ ;
    if ( i <<= 5 )
    {
        printf ( "\nC adds wings to your thoughts" ) ;
        exit( ) ;
        main( ) ;
}
```

Output:

Closing brace of if missing

(b) main( )

```
{
    static int i = 0 ;
    i++ ;
    if ( i <<= 5 )
    {
        printf ( "\n%d", i ) ;
        main( ) ;
    }
```

```
    else
        exit( ) ;
}
```

Output:

```
1
2
3
4
5
```

[J] Attempt the following:

(a) A 5 digit positive integer is entered through the keyboard, write a function to calculate sum of digits of the 5 digit number:

(1) Without using recursion
(2) Using recursion

Program:

```
/* Calculate sum of digits of a five-digit number with/without recursion */

#include <stdio.h>
#include <conio.h>

main( )
{
    int s, rs ;
    int n ;

    clrscr( ) ;
    printf ( "Enter number" ) ;
    scanf ( "%d", &n ) ;

    s = sum ( n ) ; /* Function call without recursion */
```



```
    printf ( "\nSum digits without using recursion is %d", s ) ;
    rs = rsum ( n ) ; /* Function call with recursion */
    printf ( "\n\nSum of digits using recursion is %d", rs ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

sum ( int num ) /* Function without recursion */
{
    int remainder, sum = 0 ;

    /* This time our code is very short because we can now use a while"
     clause which was not used in the earlier instance of this program */

    while ( num > 0 )
    {
        remainder = num % 10 ; /* Calculate remainder */
        sum = sum + remainder ; /* update sum */
        num = num / 10 ; /* Remove last digit */
    }
    return ( sum ) ;
}

rsum ( int num ) /* Function with recursion */
{
    int sum = 0 ;
    int remainder ;

    if ( num != 0 )
    {
        remainder = num % 10 ; /* Calculate remainder */
        sum = remainder + rsum( num/10 ) ; /* Recursive call */
    }

    return sum ;
}
```

(b) A positive integer is entered through the keyboard, write a program to obtain the prime factors of the number. Modify the function suitably to obtain the prime factors recursively.

Program:

```
/* Find Prime Factors of a number recursively */

#include <stdio.h>
#include <conio.h>

main( )
{
    int  num ;

    clrscr( ) ;
    printf ( "Enter a number" ) ;
    scanf ( "%d", &num ) ;

    printf ( "Prime factors are:" ) ;
    factor ( num ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

factor ( int n )
{
    static  int i = 2 ;

    if ( i <= n )
    {
        if ( n % i == 0 )
        {
            printf ( "%d ", i ) ;
            n = n / i ;
        }
```



```
        else
            i++ ;

        factor ( n ) ;  /* Recursive call */
    }
    return ;
}
```

(c) Write a recursive function to obtain the first 25 numbers of a Fibonacci sequence. In a Fibonacci sequence the sum of two successive terms gives the third term. Following are the first few terms of the Fibonacci sequence:

1 1 2 3 5 8 13 21 34 55 89....

Program:

```
/* Generate first 25 terms of a fibonacci sequence */

#include <stdio.h>
#include <conio.h>

main( )
{
    int  i, t,  old = 0, current = 1, new ;

    clrscr( ) ;
    printf ( "%d\t%d\t", old, current ) ;
    fibo ( old, current ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

fibo ( int old, int current )
{
    static int  terms = 2 ;
```


www.indianfun.org

```
    int new ;

    if ( terms < 20 )
    {
        new = old + current ;

        printf ( "%d\t", new ) ;
        terms = terms + 1 ;
        fibo ( current, new ) ;
    }
    else
        return ;
}
```


Chapter 6
Data Types Revisited

[A] What will be the output of the following programs:

(a)
```
main( )
{
    int i ;
    for ( i = 0 ; i <<= 50000 ; i++ )
        printf ( "\n%d", i ) ;
}
```

Output:

Program will get stuckup in infinite loop

(b)
```
main( )
{
    float a = 13.5 ;
    double b = 13.5 ;
    printf ( "\n%f %lf", a, b ) ;
}
```

Output:

13.500000 , 13.500000

(c)
```
int i = 0 ;
main( )
{
```

```
    printf ( "\nmain's i = %d", i ) ;
    i++ ;
    val( ) ;
    printf ( "\nmain's i = %d", i ) ;
    val( ) ;
}

val( )
{
    i = 100 ;
    printf ( "\nval's i = %d", i ) ;
    i++ ;
}
```

Output:

```
main's i = 0
val's i = 100
main's i = 101
val's i = 100
```

(d)

```
main( )
{
    int x, y, s = 2 ;
    s *= 3 ;
    y = f ( s ) ;
    x = g ( s ) ;
    printf ( "\n%d %d %d", s, y, x ) ;
}
int t = 8 ;
f ( int a )
{
    a += -5 ;
    t -= 4 ;
    return ( a + t ) ;
}
g ( int a )
```


www.indianfun.org


```
{
    a = 1 ;
    t += a ;
    return ( a + t ) ;
}
```

Output:

```
6 5 6
```

(e)

```
main( )
{
    static int count = 5 ;
    printf ( "\ncount = %d", count-- ) ;
    if ( count != 0 )
        main( ) ;
}
```

Output:

```
5 4 3 2 1
```

(f)

```
main( )
{
    int i, j ;
    for ( i = 1 ; i << 5 ; i++ )
    {
        j = g ( i ) ;
        printf ( "\n%d", j ) ;
    }
}

g ( int x )
{
    static int v = 1 ;
    int b = 3 ;
    v += x ;
```

www.indianfun.org

```
    return ( v + x + b ) ;
}
```

Output:

```
6  9  13  18
```

(g)
```
float  x = 4.5 ;
main( )
{
    float  y, float  f( ) ;
    x *= 2.0 ;
    y = f ( x ) ;
    printf ( "\n%f %f", x, y ) ;
}

float f ( float a )
{
    a += 1.3 ;
    x -= 4.5 ;
    return ( a + x ) ;
}
```

Output:

```
4.500000  5.800000
```

[B] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    long  num ;
    num = 2 ;
    printf ( "\n%ld", num ) ;
}
```

No Error

(b)
```
main( )
{
    char  ch = 200 ;
    printf ( "\n%d", ch ) ;
}
```

No Error

(c)
```
main( )
{
    unsigned  a = 25 ;
    long unsigned  b = 25l ;
    printf ( "\n%lu %u", a, b ) ;
}
```

Error. Format %lu should be used for b and %u should be used for a.

(d)
```
main( )
{
    long float  a = 25.345e454 ;
    unsigned double  b = 25 ;
    printf ( "\n%lf %d", a, b ) ;
}
```

Error. Format %lf should be used for b.

(e)
```
main( )
{
    float a = 25.345 ;
    float *b ;
    b = &a ;
    printf ( "\n%f %u", a, b ) ;
}
```

No Error

[C] State whether the following statements are True or False:

(a) Storage for a register storage class variable is allocated each time the control reaches the block in which it is present.

Answer: True

(b) Most of the times we need to use automatic storage class variables.

Answer: True

(c) An extern storage class variable is not available to the functions that precede its definition, unless the variable is explicitly declared in these functions.

Answer: True

(d) The value of an automatic storage class variable persists between various function invocations.

Answer: False

(e) If the CPU registers are not available, the register storage class variables are treated as static storage class variables.

Answer: False

(f) The register storage class variables cannot hold float values.

Answer: True

[D] Following program calculates the sum of digits of the number 12345. Go through it and find out why is it necessary to declare the storage class of the variable **sum** as **static**.

```
main( )
{
    int a ;
    a = sumdig ( 12345 ) ;
    printf ( "\n%d", a ) ;
}

sumdig ( int num )
{
    static int sum ;
    int a, b ;

    a = num % 10 ;
    b = ( num - a ) / 10 ;
    sum = sum + a ;
    if ( b != 0 )
        sumdig ( b ) ;
    else
        return ( sum ) ;
}
```

Answer:

Since we are finding recursive sum of digits, sum should be initialised to 0 only once. By declaring variable sum as static, the statement

```
static int sum = 0 ;
```

would be executed only once ensuring that slum gets a value 0 only during the first call to the function sumdig( )

Chapter 7
The C Preprocessor

[A] Answer the following:

(a) What is a preprocessor directive

1. a message from compiler to the programmer
2. a message from compiler to the linker
3. a message from programmer to the preprocessor
4. a message from programmer to the microprocessor

Answer:

3. a message from programmer to the preprocessor

(b) Which of the following are correctly formed **#define** statements

```
#define  INCH PER FEET 12
#define  SQR(X) (X*X)
#define  SQR(X)  X*X
#define  SQR(X) (X*X)
```

Answer:

```
#define SQR(X)(X*X)
```

(c) State True or False:

1. A macro must always be written in capital letters.

Answer: False

2. A macro should always be accomodated in a single line.

Answer: Usually true. Some compilers allow multi-line macros

3. After preprocessing when the program is sent for compilation the macros are removed from the expanded source code.

Answer: True

(d) How many **#include** directives may be there in a given program file?

Answer:

As many as you desire

(e) What is the difference between the following two **#include** directives:

```
#include "conio.h"
#include <conio.h>
```

Answer:

#include "conio.h": This command would look for the file conio.h in the current directory as well as the specified list of directories as mentioned in the search path that might have been set up.

#include<conio.h>: This command would look for the file conio.h in the specified list of directories only.

(f) A header file is:

1. A file that contains standard library functions
2. A file that contains definitions and macros
3. A file that contains user defined functions
4. A file that is present in current working directory

Answer:

2. A file that contains definitions and macros

[B] What will be the output of the following program:

(a)
```
main( )
{
    int i = 2 ;
    #ifdef DEF
        i *= i ;
    #else
        printf ( "\n%d", i ) ;
    #endif
}
```

Output:

2

(b)
```
#define PRODUCT(x) ( x * x )
main( )
{
    int i = 3, j ;
    j = PRODUCT( i + 1 ) ;
    printf ( "\n%d", j ) ;
}
```

Output:

7

(c)
```
#define PRODUCT(x) ( x * x )
main( )
{
    int i = 3, j, k ;
    j = PRODUCT( i++ ) ;
    k = PRODUCT ( ++i ) ;

    printf ( "\n%d %d", j, k ) ;
}
```

Output:

9 49

[C] Attempt the following:

(a) Write down macro definitions for the following:

(1) To test whether a character entered is a small case letter or not.
(2) To test whether a character entered is a upper case letter or not.
(3) To test whether a character is an alphabet or not. Make use of the macros you defined in (1) and (2) above.
(4) To obtain the bigger of two numbers.

Program:

```
/* Macros like ISUPPER, ISLOWER, ISAPLHA, BIG */

#include <stdio.h>
#include <conio.h>

/* Macros Defined Below */
#define ISUPPER(x) ( x >= 65 && x <= 90 ? 1 : 0 )
#define ISLOWER(x) ( x >= 97 && x <= 122 ? 1 : 0 )
```



```
#define ISALPHA(x) ( ISUPPER(x) || ISLOWER(x) )
#define BIG(x,y) ( x > y ? x : y )

main( )
{
    char ch ;
    int d, a, b ;

    clrscr( ) ;
    printf ( "Enter any alphabet/character:" ) ;
    scanf ( "%c", &ch ) ;

    if ( d = ISUPPER ( ch ) == 1 ) /* Macro substitution */
        printf ( "\nYou entered a capital letter" ) ;
    else if ( d = ISLOWER ( ch ) == 1 ) /* Macro substitution */
        printf ( "\nYou entered a small case letter" ) ;
    else if ( d = ISALPHA ( ch ) != 1 ) /* Macro substitution */
        printf ( "\nYou entered character other than an alphabet" ) ;

    printf ( "\n\nEnter any two numbers " ) ;
    scanf ( "%d%d", &a, &b ) ;

    d = BIG ( a, b ) ; /* Macro substitution */
    printf ( "\nBigger number is %d", d ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(b) Write macro definitions with arguments for calculation of area and perimeter of a triangle, a square and a circle. Store these macro definitions in a file called "areaperi.h". Include this file in your program, and call the macro definitions for calculating area and perimeter for different squares, triangles and circles.

Program:

```
/* Calculation of area, perimeter and circumference using macros */

#include <stdio.h>
#include <conio.h>

/* include file containing macro definitions */

#include "areaperi.h"

main( )
{
    int d, a, b ;
    float sid1, sid2, sid3, sid, p_tri, p_cir, p_sqr, a_tri, a_cir,a_sqr ;
    float r, base, height ;

    clrscr( ) ;
    printf ( "\nEnter radius of circle" ) ;
    scanf ( "%f", &r ) ;
    p_cir = PERIC ( r ) ;
    printf ( "Circumference of circle = %f", p_cir ) ;
    a_cir = AREAC ( r ) ;
    printf ( "\nArea of circle = %f", a_cir ) ;

    printf ( "\n\nEnter side of a square " ) ;
    scanf ( "%f", &sid ) ;
    p_sqr = PERIS ( sid ) ;
    printf ( "Perimeter of square = %f", p_sqr ) ;
    a_sqr = AREAS ( sid ) ;
    printf ( "\nArea of square = %f", a_sqr ) ;

    printf ( "\n\nEnter length of 3 sides of triangle " ) ;
    scanf ( "%f %f %f", &sid1, &sid2, &sid3 ) ;
    p_tri = PERIT ( sid1, sid2, sid3 ) ;
```



```
    printf ( "Perimeter of triangle = %f", p_tri ) ;

    printf ( "\n\nEnter base and height of triangle " ) ;
    scanf ( "%f %f", &base, &height ) ;
    a_tri = AREAT ( base, height ) ;
    printf ( "Area of triangle = %f", a_tri ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

Chapter 8

Arrays

Simple arrays

[A] What will be the output of the following programs:

(a)
```
main( )
{
    int num[26], temp ;
    num[0] = 100 ;
    num[25] = 200 ;
    temp = num[25] ;
    num[25] = num[0] ;
    num[0] = temp ;
    printf ( "\n%d %d", num[0], num[25] ) ;
}
```

Output:

200 100

(b)
```
main( )
{
    int array[26], i ;
    for ( i = 0 ; i <= 25 ; i++ )
    {
        array[i] = 'A' + i ;
        printf ( "\n%d %c", array[i], array[i] ) ;
    }
```

```
}
```

Output:

```
65  A
66  B
.....
.....
90  Z
```

(c)
```
main( )
{
    int sub[50], i ;
    for ( i = 0 ; i <= 48 ; i++ )
    {
        sub[i] = i ;
        printf ( "\n%d", sub[i] ) ;
    }
}
```

Output:

(a) 49. Since I takes a value 49 when it reaches the statement
(b) sub[I] = I ;

[B] Point out the errors, if any, in the following program segments:

(a)
```
/* mixed has some char and some int values */
int char mixed[100] ;
```

Error. An array cannot be declared to be of the type in char

(b)
```
main( )
{
    int a[10], i ;

    for ( i = 1 ; i <= 10 ; i++ )
    {
        scanf ( "%d", a[i] ) ;
        printf ( "%d", a[i] ) ;
    }
}
```

Error. Array bounds are exceeded

(c)

(c)
```
main( )
{
    int size ;
    scanf ( "%d", &size ) ;
    int arr[size] ;
    for ( i = 1 ; i <= size ; i++ )
    {
        scanf ( "%d", arr[i] ) ;
        printf ( "%d", arr[i] ) ;
    }
}
```

Error. Dimension of the array should be constant and all declarations should be at the beginning.

[C] Answer the following:

(a) An array is a collection of

1. different data types scattered throughout memory
2. the same data type scattered throughout the memory
3. the same data type placed next to each other in memory
4. different data types placed next to each other in memory

Answer:

3. the same data type placed next to each other in memory

(b) Is this a correct array declaration?

int num (25) ;

Answer:

No .Use [ ] in place of ( )

(c) Which element of the array does this expression reference?

num[4]

Answer:

Fifth element

(d) What is the difference between the 5's in these two expressions? (select the correct Answer)

```
int num[5] ;
num[5] = 11 ;
```

1. first is particular element, second is type
2. first is array size, second is particular element
3. first is particular element, second is array size
4. both specify array size

Answer:

2. first is array size, second is particular element

(e) State whether the following statements are True or False:

1. The array **int num[26]** has twenty-six elements.

Answer: True

2. The expression **num[1]** designates the first element in the array

Answer: False

3. The expression **num[27]** designates the twenty-eighth element in the array.

Answer: True

[D] Attempt the following:

(a) Twenty five numbers are entered from the keyboard into an array. Write a program to find out how many of them are positive, how many are negative, how many are even and how many odd.

Program

```
/* Program to count positive, negative, odd & even numbers in an
array */

#include <stdio.h>
#include <conio.h>

main ( )
{
    int  num[25], i, neg = 0, pos = 0, odd = 0, even = 0 ;

    clrscr( ) ;
    printf ( "Enter 25 elements of array" ) ;
    for ( i = 0 ; i <= 24 ; i++ )
        scanf ( "%d", &num[i] ) ;  /* Array Elements */
```

```
    for ( i = 0 ; i <= 24 ; i++ )
    {
        num[i] < 0 ? neg++ : ( pos++ ) ;  /* conditional operators */
        num[i] % 2 ? odd++ : ( even++ ) ;
    }
    printf ( "\nNegative elements = %d", neg ) ;
    printf ( "\nPositive elements = %d", pos ) ;
    printf ( "\nEven elements = %d", even ) ;
    printf ( "\nOdd elements = %d", odd ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(b) Implement the Selection Sort, Bubble Sort and Insertion sort algorithms on a set of 25 numbers. (Refer Figure 8.1 for the logic of the algorithms)

- Selection sort
- Bubble Sort
- Insertion Sort

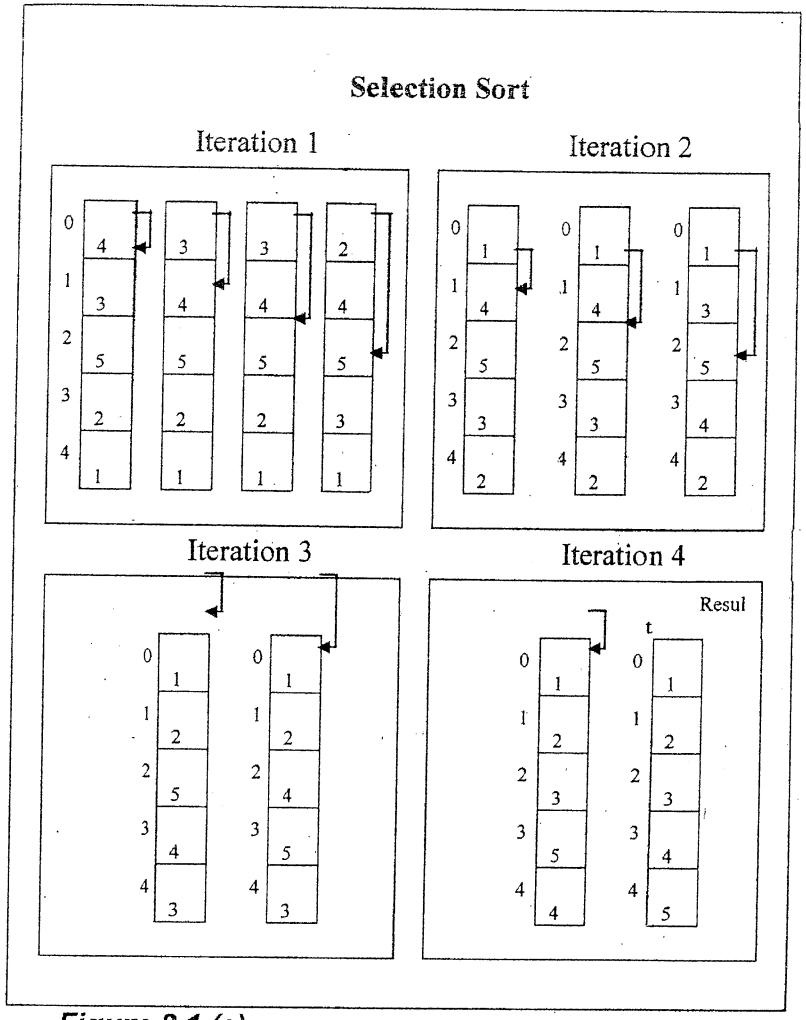


Figure 8.1 (a)

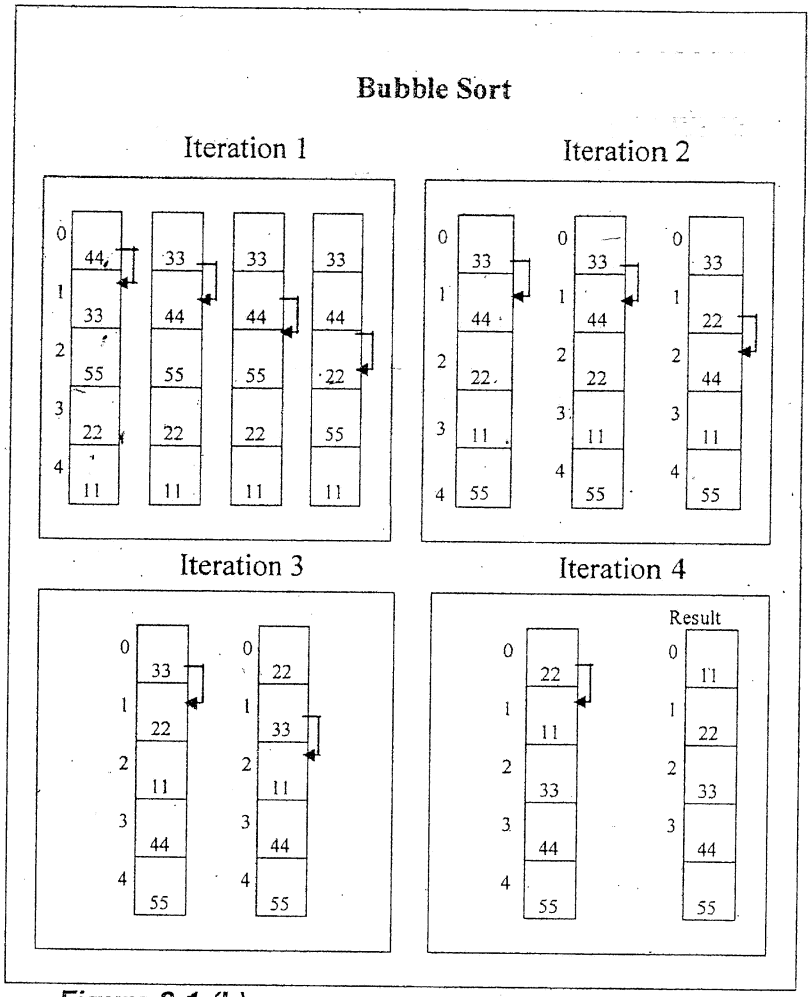


Figure 8.1 (b)

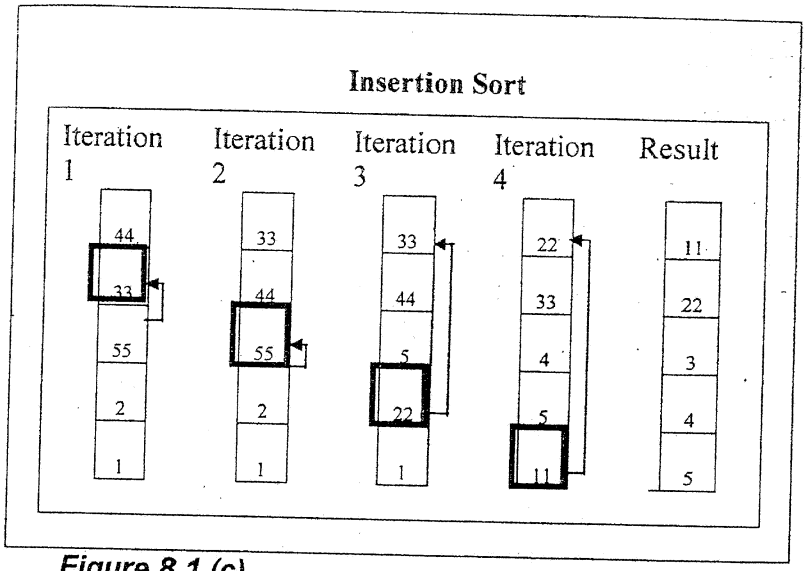


Figure 8.1 (c)

Program

```
/* Selection Sort */

#include <stdio.h>
#include <conio.h>

main ( )
{
    int  a[25], i, j, t ;

    clrscr( ) ;
    printf ( "\nEnter 25 numbers: " ) ;

    for ( i = 0 ; i <= 24 ; i++ )
        scanf ( "%d", &a[i] ) ;  /* Enter Array Elements */

    for ( i = 0 ; i <= 23 ; i++ )  /* Number of passes */
```

```
    {
        for ( j = i + 1 ; j <= 24 ; j++ )  /* start at next element */
        {
        /* compare an element a[i] with all elements a[j] one by one */
            if ( a[i] > a[j] )
            {
                t = a[i] ;  /* Interchange if greater than next element */
                a[i] = a[j] ;
                a[j] = t ;
            }
        }
    }
    printf ( "\nSorted Numbers are:\n" ) ;
    for ( i = 0 ; i <= 24 ; i++ )
        printf ( "%d ", a[i] ) ;  /* print the sorted array */

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

/* Exchange Sort */

```
#include <stdio.h>
#include <conio.h>

main( )
{
    int  a[25], i, j, m, t ;

    clrscr( ) ;
    printf ( "\nEnter 25 numbers:" ) ;
    for ( i = 0 ; i <= 24 ; i++ )
        scanf ( "%d", &a[i] ) ;  /* Enter array elements */

    for ( i = 0 ; i <= 24 ; i++ )  /* Number of passes */
    {
        for ( j = 0 ; j < 24-i ; j++ )  /* start another loop from begining */
        {
```



```
            /* compare an element with the next element one by one */
            if ( a[j] > a[j+1] )
            {
                t = a[j] ;  /* Interchange the two if first is greater */
                a[j] = a[j+1] ;
                a[j+1] = t ;
            }
        }
    }
    printf ( "\nSorted Numbers are:\n" ) ;
    for ( i = 0 ; i <= 24 ; i++ )
        printf ( "%d", a[i] ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

/* Insertion Sort */

```
#include <stdio.h>
#include <conio.h>

main( )
{
    int  a[25], i, j, k, t ;

    clrscr( ) ;
    printf ( "\nEnter 25 Numbers:\n" ) ;
    for ( i = 0 ; i <= 9 ; i++ )
        scanf ( "%d", &a[i] ) ;  /* Enter array elements */

    for ( i = 1 ; i <= 9 ; i++ )  /* Number of passes */
    {
        t = a[i] ;
        for ( j = 0 ; j < i ; j++ )
        {
```

```
            if ( t < a[j] )
            {
                for ( k = i ; k >= j ; k-- )
                    a[k] = a[k-1] ; /* Interchange elements */
                a[j] = t ;
                break ;
            }
        }
    }
    printf ( "\nSorted Numbers are:\n" ) ;
    for ( i = 0 ; i <= 9 ; i++ )
        printf ( "%d ", a[i] ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(c) Implement the following procedure to generate prime numbers from 1 to 100 into a program. This procedure is called sieve of Eratosthenes.

step 1	Fill an array **num[100]** with numbers from 1 to 100
step 2	Starting with the second entry in the array, set all its multiples to zero.
step 3	Proceed to the next non-zero element and set all its multiples to zero.
step 4	Repeat step 3 till you have set up the multiples of all the non-zero elements to zero
step 5	At the conclusion of step 4, all the non-zero entries left in the array would be prime numbers, so print out these numbers.

Program:

```
/* Sieve of Eratosthenes */
```



```
#include <stdio.h>
#include <conio.h>

main( )
{
    int num[100], i, j, k, step ;

    clrscr( ) ;

    for ( i = 0 ; i <= 99 ; i++ )
        num[i] = i + 1 ; /* Fill array with numbers from 1 to 100 */

    for ( i = 1 ; i <= 99 ; i++ )
    {
        if ( num[i] != 0 )
        {
            k = num[i] * 2 - 1 ;
            step = num[i] ;
            for ( j = k ; j <= 99 ; j = j + step )
                num[j] = 0 ;
        }
    }
    printf ( "\nPrime numbers between 1 & 100 are:\n" ) ;

    for ( i = 0 ; i <= 99 ; i++ )
    {
        if ( num[i] != 0 )
            printf ( "%d\t", num[i] ) ;
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

More on arrays, Arrays and pointers

[E] What will be the output of the following programs:

(a) main( )

```
{
    int a[10] ;
    for ( i = 1 ; i <= 10 ; i++ )
    {
        scanf ( "%d", a[i] ) ;
        printf ( "%d", a[i] ) ;
    }
}
```

Output:

```
10
20
30
40
50
```

(b) main( )

```
{
    int b[ ] = { 0, 20, 0, 40, 5 } ;
    int i, *k ;
    k = b ;
    for ( i = 0 ; i <= 4 ; i++ )
    {
        printf ( "\n%d" *k ) ;
        k++ ;
    }
}
```

Output:

```
0
20
0
40
```



```
5
```

(c) main( )

```
{
    int a[ ] = { 2, 4, 6, 8, 10 } ;
    int i ;
    change ( a, 5 ) ;
    for ( i = 0 ; i <= 4 ; i++ )
        printf ( "\n%d", a[i] ) ;
}
change ( int *b, int n )
{
    int i ;
    for ( i = 0 ; i < n ; i++ )
        *( b + i ) = *( b + i ) + 5 ;
}
```

Output:

```
7
9
11
13
15
```

(d) main( )

```
{
    int a[5], i, b = 16 ;
    for ( i = 0 ; i < 5 ; i++ )
        a[i] = 2 * i ;
    f ( a, b ) ;
    for ( i = 0 ; i < 5 ; i++ )
        printf ( "\n%d", a[i] ) ;
    printf ( "\n%d", b ) ;
}
f ( int *x, int y )
```

```
{
    int i ;
    for ( i = 0 ; i < 5 ; i++ )
        *( x + i ) += 2 ;
    y += 2 ;
}
```

Output:

```
2
4
6
8
10
16
```

(e)
```
main( )
{
    static int a[5] ;
    int i ;
    for ( i = 0 ; i <= 4 ; i++ )
        printf ( "\n%d", a[i] ) ;
}
```

Output:

```
0
0
0
0
0
```

(f)
```
main( )
{
    int a[5] = { 5, 1, 15, 20, 25 } ;
    int i, j, k = 1, m ;
    i = ++a[1] ;
```



```
    j = a[1]++ ;
    m = a[i++] ;
    printf ( "\n%d %d %d", i, j, m ) ;
}
```

Output:

```
3  2  15
```

[F] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    int array[6] = { 1, 2, 3, 4, 5, 6 } ;
    int i ;
    for ( i = 0 ; i <= 25 ; i++ )
        printf ( "\n%d", array[i] ) ;
}
```

No Error. Caution: Array bounds are being exceeded.

(b)
```
main( )
{
    int sub[50], i ;
    for ( i = 1 ; i <= 50 ; i++ )
    {
        sub[i] = i ;
        printf ( "\n%d" , sub[i] ) ;
    }
}
```

No Error. Caution: Array bounds are being exceeded.

(c)
```
main( )
{
    int a[ ] = { 10, 20, 30, 40, 50 } ;
    int j ;
```

```
    j = a ; /* store the address of zeroth element */
    j = j + 3 ;
    printf ( "\n%d" *j ) ;
}
```

Error. It should be int * j.

(d) main( )

```
{
    float a[ ] = { 13.24, 1.5, 1.5, 5.4, 3.5 } ;
    float *j ;
    j = a ;
    j = j + 4 ;
    printf ( "\n%d %d %d", j, *j, a[4] ) ;
}
```

No Error. Use %f for printing out *j and a[4].

(e) main( )

```
{
    float a[ ] = { 13.24, 1.5, 1.5, 5.4, 3.5 } ;
    float *j, *k ;
    j = a ;
    k = a + 4 ;
    j = j * 2 ;
    k = k / 2 ;
    printf ( "\n%d %d", *j, *k ) ;
}
```

Error. Multiplication and division operations are not allowed on pointers.

(f) main( )

```
{
    int max = 5 ;
    float arr[max] ;
```



```
    for ( i = 0 ; i < max ; i++ )
        scanf ( "%f", &arr[i] ) ;
}
```

Error. Array dimension cannot be variable.

[G] Answer the following:

(a) What will happen if you try to put so many values into an array when you initialise it that the size of the array is exceeded?

1. nothing
2. possible system malfunction
3. Error message from the compiler
4. other data may be overwritten

Answer:

2. possible system malfunction

(b) In an array **int arr[12]** the word **arr** represents the a address of the array

(c) What will happen if you put too few elements in an array when you initialise it?

1. nothing
2. possible system malfunction
3. Error message from the compiler
4. unused elements will be filled with 0's or garbage

Answer:

4. unused elements would be filled with 0's or garbage

(d) What will happen if you assign a value to an element of an array whose subscript exceeds the size of the array?

1. the element will be set to 0
2. nothing, it's done all the time
3. other data may be overwritten
4. Error message from the compiler

Answer:

3. other data may be overwritten

(e) When you pass an array as an argument to a function, what actually gets passed?

1. address of the array
2. values of the elements of the array
3. address of the first element of the array
4. number of elements of the array

Answer:

1. address of the array

(f) Which of these are reasons for using pointers?

1. To manipulate parts of an array
2. To refer to keywords such as for and if
3. To return more than one value from a function
4. To refer to particular programs more conveniently

Answer:

1. To manipulate parts of array
3. To return more than one value from a function

(g) If you don't initialise a static array, what will be the elements set to?

1. 0
2. an undetermined value
3. a floating point number
4. the character constant '\0'

Answer:

1. 0

(h) State True or False:

Address of a floating point variable is always a whole number.

Answer: True

(i) Which of the following is the correct way of declaring a float pointer:

1. float ptr ;
2. float *ptr ;
3. *float ptr ;
4. None of the above

Answer:

2. float *ptr

(j) Add the missing statement for the following program to print 35.

```
main( )
{
    int j, *ptr ;
    *ptr = 35 ;
    printf ( "\n%d", j ) ;
}
```

Answer:

```
ptr = & j
```

(k) if **int s[5]** is a one dimensional array of integers, which of the following refers to the third element in the array?

1. *(s + 2)
2. *(s + 3)
3. s + 3
4. s + 2

Answer:

1. * (s + 2)

[H] Attempt the following:

(a) Write a program which performs the following tasks:

- initialise an integer array of 10 elements in **main( )**
- pass the entire array to a function **modify( )**
- in modify( ) multiply each element of array by 3
- return the control to **main( )** and print the new array elements in **main( )**

Program:

```
/* Program to pass the entire array and multiply each element by 3 */

#include <stdio.h>
#include <conio.h>
main( )
{
    int i ;
    /* Initialise an array of ten elements */
    static int  array[ ] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 } ;

    clrscr( ) ;
    printf ( "\nOriginal Array is:\n" ) ;
```



```
    for ( i = 0 ; i < 10 ; i++ )
        printf ( "%d ", array[i] ) ;  /* Print original array */

    modify ( array, 10 ) ;  /* Array passed to function */
    /* This will pass the base address of the array */

    printf ( "\n\nModified Array is: \n" ) ;
    for ( i = 0 ; i < 10 ; i++ )
        printf ( "%d ", array[i] ) ;  /* Print modified array */

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
/* Function to modify array */
modify ( int *arr, int n )  /* Pointer variable since address is being
                               passed */
{
    int i ;
    for ( i = 0 ; i < n ; i++ )
    {
        *arr = *arr * 3 ;  /* Replace array elements by new elements */
        arr++ ;  /* advance to address of next element */
    }
}
```

(b) The screen is divided into 25 rows and 80 columns. The characters which are displayed on the screen are stored in a special memory called VDU memory (not to be confused with ordinary memory). Each character displayed on the screen occupies two bytes in VDU memory. The first of these bytes contains the ascii value of the character being displayed whereas the second byte contains the colour in which the character is displayed.

For example, the ascii value of the character present on zeroth row and zeroth column on the screen is stored at location number 0xB8000000. Therefore the colour of this character

would be present at location number 0xB8000000 + 1. Similarly ascii value of character in row 0, col 1 will be at location 0xB8000000 + 2, and its colour at 0xB8000000 + 3. With this knowledge write a program which when executed would keep converting every capital letter on the screen to small case letter and every small case letter to capital letter. The procedure should stop the moment the user hits a key from the keyboard.
This is an activity of a rampant Virus called Dancing Dolls. (For monochrome adapter, use 0xB0000000 instead of 0xB8000000).

Program:

```
/* Dancing Dolls */

#include <stdio.h>
#include <dos.h>

main( )
{
    char far *scr = 0xB8000000 ;  /* Far pointer pointing to base
                                     address of VDU memory */
    int i ;

    while ( !kbhit( ) )  /* check whether key is hit */
    {
        /* No of Bytes of VDU memory = 80 x 25 x 2 = 4000 */
        for( i = 0 ; i <= 3999 ; i += 2 )
        if ( scr[i]  >= 'A' && scr[i] <= 'Z' )
        {
            scr [i]  += 32 ;
        }
        else
        {
            if ( scr[i] >= 'a' && scr[i]  <= 'z' )
            {
```



```
                scr[i]  -= 32 ;
            }
        }
    }
}
```

More than one dimension

[I] What will be the output of the following programs:

(a)
```
main( )
{
    int  n[3][3] = {
                        2, 4, 3,
                        6, 8, 5,
                        3, 5, 1
                    };
    printf ( "\n%d %d %d", *n, n[3][3], n[2][2] ) ;
}
```

Output:

<base address><garbage value> 1

(b)
```
main( )
{
    int n[3][3] = {
                        2, 4, 3,
                        6, 8, 5,
                        3, 5, 1
                    };
    int  i, *ptr ;
    ptr = n ;
    for ( i = 0 ; i <= 8 ; i++ )
        printf ( "\n%d", *( ptr + i ) ) ;
}
```

Output:

```
2
4
3
6
8
5
3
5
1
```

(c)
```
main( )
{
    int n[3][3] = {
                    2, 4, 3,
                    6, 8, 5,
                    3, 5, 1
                  };
    int i, j ;
    for ( i = 0 ; i <= 2 ; i++ )
        for ( j = 0 ; j <= 2 ; j++ )
            printf ( "\n%d %d", n[i][j], *( *( n + i ) + j ) ) ;
}
```

Output:

```
2  2
4  4
3  3
6  6
.....
....
1  1
```

[J] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    int twod[ ][ ] = {
                        2, 4,
                        6, 8
                      };
    printf ( "\n%d", twod ) ;
}
```

Error. While declaring a two dimensional array mentioning the column dimension is necessary

(b)
```
main( )
{
    int three[3][ ] = {
                        2, 4, 3,
                        6, 8, 2,
                        2, 3 ,1
                      };
    printf ( "\n%d", three[1][1] ) ;
}
```

Error. While declaring a two dimensional array mentioning the column dimension is necessary

[K] How will you initialise a three dimensional array **threed[3][2][3]**? How will you refer the first and last element in this array?

Answer:

```
int arr[3][2][3] = {
                     {
                         {1,2,3},
                         {4,5,6}
                     },
                     {
```

```
                {1,2,3},
                {4,5,6}
            },
            {
                {1,2,3},
                {4,5,6}
            }
        };
```

The first element of the array is arr**[0][0][0]** **or** ***arr**
The last element of the array is arr[2][1][2] or *(*(*(arr + 2) + 1) + 2)

[L] Attempt the following:

(a) Write a program to pick up the largest number from any 5 row by 5 column matrix.

Program:

```
/* Pick up largest number from 5 x 5 matrix */

#include <stdio.h>
#include <conio.h>

main( )
{
   /* Intialise 5 x 5 Matrix */
   /* Instead of initialising a matrix, it can also be interactively entered
      from the keyboard, if required */
   static int  a[5][5] = {
                          { 11,  1,  7,  9, 7 },
                          { 13, 54, 56,  2, 5 },
                          { 23, 43, 89, 22, 13 },
                          { 14, 15, 17, 16, 19 },
                          { 45,  3,  6,  8, 10 }
                        };
```



```
   int  i, j, big ;

   clrscr( ) ;
   big = a[0][0] ;
   printf ( "\nThe 5 x 5 matrix is:\n" ) ;

   for ( i = 0 ; i <= 4 ; i++ )
   {
      for ( j = 0 ; j <= 4 ; j++ )
      {
         printf ( "%d\t" , a[i][j] ) ;  /* Print the row of the matrix */
         if ( a[i][j] > big )
            big = a[i][j] ;
      }
   printf ( "\n" ) ;  /* Newline for next row */

   }
   printf ( "\nLargest number in the matrix is %d", big ) ;
   printf ( "\n\n\n\n\nPress any key to exit..." ) ;
   getch( ) ;
}
```

(b) Write a program to obtain transpose of a 4 x 4 matrix. The transpose of a matrix is obtained by exchanging the elements of each row with the elements of the corresponding column.

Program:

```
/* Transpose of a 4 x 4 matrix */

#include <stdio.h>
#include <conio.h>

main( )
{
   int  mat[4][4], i, j, temp ;
```

```
clrscr( ) ;
printf ( "\nEnter values for 4 x 4 matrix:\n " ) ;

for ( i = 0 ; i <= 3 ; i++ )
{
     for ( j = 0 ; j <= 3 ; j++ )
     {
          scanf ( "%d", &mat[i][j] ) ; /* get values for the matrix */
     }
}

/* print the matrix as entered */
printf ( "\n\nThe matrix you entered is:\n" ) ;

for ( i = 0 ; i <= 3 ; i++ )
{
     for ( j = 0 ; j <= 3 ; j++ )
          printf ( "%d\t", mat[i][j] ) ;

     printf ( "\n" ) ;
}

/* Transpose the matrix */
for ( i = 0 ; i <= 3 ; i++ )
{
     for ( j = i + 1 ; j <= 3 ; j++ )
     {
          temp = mat[i][j] ;
          mat[i][j] = mat[j][i] ; /* row interchanged with column */
          mat[j][i] = temp ;
     }
}
/* Print the transposed matrix */
printf ( "\n\nTranspose of the matrix is:\n" ) ;

for ( i = 0 ; i <= 3 ; i++ )
{
```



```
          for ( j = 0 ; j <= 3 ; j++ )
               printf ( "%d\t", mat[i][j] ) ;
          printf ( "\n" ) ;
     }
     printf ( "\n\n\n\n\nPress any key to exit..." ) ;
     getch( ) ;
}
```

(c) Very often in fairs we come across a puzzle, which contains 15 numbered square pieces, which can be moved horizontally or vertically. A possible arrangement of these pieces is shown below:

1	4	15	7
8	10	2	11
14	3	6	13
12	9	5	

Figure 8.2

As you can see there is a blank at bottom right corner. Implement the following procedure through a program:

Draw the boxes as shown above. Display the numbers in the above order. Allow the user to hit any of the arrow keys (up, down, left, or right).

If user hits say, right arrow key then the piece with a number 5 should move to the right and blank should replace the original position of 5. Similarly, if down arrow key is hit, then 13 should move down and blank should replace the original position of 13. If left arrow key or up arrow key is hit then no action should be taken.

The user would continue hitting the arrow keys till the numbers aren't arranged in ascending order.

Keep track of the number of moves in which the user manages to arrange the numbers in ascending order. The user who manages it in minimum number of moves is the one who wins.

How do we tackle the arrow keys? We cannot receive them using **scanf( )** function. Arrow keys are special keys which are identified by their 'scan codes'. Use the following function in your program. It would return the scan code of the arrow key being hit. Don't worry about how this function is written. We are going to deal with it later. The scan codes for the arrow keys are:

up arrow key - 72 down arrow key - 80
left arrow key - 75 right arrow key – 77

```
/* Returns scan code of the key that has been hit */
#include <dos.h>
getkey( )
{
    union REGS i, o ;

    while ( !kbhit( ) )
        ;
    i.h.ah = 0 ;
    int86 ( 22, &i, &o ) ;
    return ( o.h.ah ) ;
}
```

Program:

```
/* Video game : Shuffle */

#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include "goto.c"

/* initialise the array for printing */
```



```
int arr[4][4] = {
                    { 1, 4, 15, 7 },
                    { 8, 10, 2, 11 },
                    { 14, 3, 6, 13 },
                    { 12, 9, 5, 0 }
                } ;
main( )
{
    int row = 3, col = 3, t, ch ;

    clrscr( ) ;
    boxes( ) ; /* function to draw boxes */
    display( ) ; /* function to display numbers */
    while( 1 )
    {
        ch = sgetkey( ) ; /* returns the scan code of the key that has
                             been hit */
        switch ( ch )
        {
            case 80 : /* Down Arrow */
                if ( row == 0 )
                {
                    printf ( "\a" ) ;
                    break ;
                }
                t = arr[row][col] ;
                arr[row][col] = arr[row -1][col] ;
                arr[row-1][col] = t ;
                row-- ;
                display( ) ;
                break ;

            case 77 : /* Right Arrow */
                if ( col == 0 )
                {
                    printf ( "\a" ) ;
                    break ;
                }
```

```
            t = arr[row][col] ;
            arr[row][col] = arr[row][col-1] ;
            arr[row][col-1] = t ;
            col-- ;
            display( ) ;
            break ;

        case 72 : /* Up Arrow */
            if ( row == 3 )
            {
                printf ( "\a" ) ;
                break ;
            }
            t = arr[row][col] ;
            arr[row][col] = arr[row + 1][col] ;
            arr[row + 1][col] = t ;
            row++ ;
            display( ) ;
            break ;

        case 75 : /* Left Arrow */
            if ( col == 3 )
            {
                printf ( "\a" ) ;
                break ;
            }
            t = arr[row][col] ;
            arr[row][col] = arr[row][col + 1] ;
            arr[row][col + 1] = t ;
            col++ ;
            display( ) ;
            break ;

        case 1 : /* Esc key */
            exit( ) ;
    }
    check( ) ; /* Prints number of moves */
}
```



```
}

check( )
{
    static int move = 0 ;
    int k = 1 , i , j ;

    move++ ;
    gotorc( 24, 30 ) ;
    printf ( "no of moves = %d" , move ) ;
    for ( i = 0 ; i <= 3 ; i++ )
    {
        for ( j = 0 ; j <= 3 ; j++ )
        {
            if ( arr[i][j] == 0 )
                continue ;
            else
                if ( arr[i][j] == k )
                    k++ ;
                else
                    return ;
        }
    }
    exit( ) ;
}

boxes( ) /* draws boxes using box drawing characters */
{
    int r , c ; /* row & column */

    clrscr( ) ;

    for ( c = 30 ; c <= 42 ; c++ )
    {
        for ( r = 8 ; r <= 16 ; r+=2 )
        {
            gotorc ( r, c ) ; /* function defined in file "goto.c" */
            printf ( "%c", 196 ) ;
```

```
        }
    }

    for ( r = 8 ; r <= 16 ; r++ )
    {
        for ( c = 30 ; c <= 42 ; c += 3 )
        {
            gotorc ( r, c ) ;
            printf ( "%c", 179 ) ;
        }
    }

    for ( c = 33 ; c <= 39 ; c += 3 )
    {
        gotorc ( 8, c ) ;
        printf ( "%c", 194 ) ;

        gotorc ( 16, c ) ;
        printf ( "%c", 193 ) ;
    }

    for ( r = 10 ; r <= 14 ; r+=2 )
    {
        gotorc ( r, 30 ) ;
        printf ( "%c", 195 ) ;

        gotorc( r , 42 ) ;
        printf ( "%c" ,180) ;
    }

    for ( r = 10 ; r <= 14 ; r+=2 )
    {
        for ( c = 33 ; c <= 39 ; c += 3 )
        {
            gotorc ( r, c ) ;
            printf ( "%c", 197 ) ;
        }
    }
```



```
        gotorc ( 8, 30 ) ;
        printf ( "%c", 218 ) ;

        gotorc ( 8, 42 ) ;
        printf ( "%c", 191 ) ;

        gotorc ( 16, 30 ) ;
        printf ( "%c", 192 ) ;

        gotorc ( 16, 42 ) ;
        printf ( "%c", 217 ) ;
    }

    display( ) /* display numbers in the boxes */
    {
        int r = 9, c = 31, i, j ;

        for ( i = 0 ; i <= 3 ; i++ )
        {
            for ( j = 0 ; j <= 3 ; j++ )
            {
                if ( arr[i][j] == 0 )
                {
                    gotorc ( r, c ) ;
                    printf ( " " ) ;
                }
                else
                {
                    gotorc ( r, c ) ;
                    printf ( "%d", arr[i][j] ) ;
                }
                c = c + 3 ;
            }
            r = r + 2 ;
            c = 31 ;
        }
    }
```

```
sgetkey( ) /* returns the scan code of the key that has been hit */
{
    union REGS i , o ;
    while ( !kbhit( ) ) ;
    i.h.ah = 0 ;
    int86 ( 22 , &i , &o ) ;
    return ( o.h.ah ) ;
}
```

(d) Those readers who are from an Engineering/Science background may try, writing programs for following problems.

(1) Write a program to add two 6 x 6 matrices.

Program:

```
/* Program to add two 6 x 6 matrices */

#include <stdio.h>
#include <conio.h>

main( )
{
   int  mat1[6][6], mat2[6][6], mat3[6][6], i, j ;

   clrscr( ) ;
   printf ( "\nEnter values for first 6 x 6 matrix:\n" ) ;
   for ( i = 0 ; i <= 5 ; i++ )
   {
        for ( j = 0 ; j <= 5 ; j++ )
             scanf ( "%d", &mat1 [i][j] ) ;  /* Enter First Matrix
                                                values */
   }

   printf ( "\nEnter values for second 6 x 6 matrix:\n" ) ;
```



```
   for ( i = 0 ; i <= 5 ; i++ )
   {
        for ( j = 0 ; j <= 5 ; j++ )
             scanf ( "%d", &mat2[i][j] ) ;  /* Enter second Matrix
                                               values */
   }

   /* print both matrices as entered */
   printf ( "\nMatrices entered by you are: \nMatrix 1:\n" ) ;
   for ( i = 0 ; i <= 5 ; i++ )
   {
        for ( j = 0 ; j <= 5 ; j++ )
             printf ( "%d\t", mat1[i][j] ) ;  /* print each row */

        printf ( "\n" ) ;  /* new row on new line */
   }

   printf ( "\nMatrix 2:\n" ) ;
   for ( i = 0 ; i <= 5 ; i++ )
   {
        for ( j = 0 ; j <= 5 ; j++ )
             printf ( "%d\t", mat2[i][j] ) ; /* print each row */

        printf ( "\n" ) ;  /* new row on new line */
   }

   /* Calculate the sum of two matrices */
   for ( i = 0 ; i <= 5 ; i++ )
   {
        for ( j = 0 ; j <= 5 ; j++ )
             mat3[i][j] = mat1[i][j] + mat2[i][j] ;
   }

   /* print the matrix with the sum */
   printf ( "\nThe new matrix after addition of the above two
             matrices is:\n" ) ;
   for ( i = 0 ; i <= 5 ; i++ )
   {
```

```
        for ( j = 0 ; j <= 5 ; j++ )
            printf ( "%d\t", mat3[i][j] ) ;

        printf ( "\n" ) ;
    }
    printf ( "\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(2) Write a program to multiply any two 3 x 3 matrices.

Program:

```
/* Program to multiply two 3 x 3 matrices */

#include <stdio.h>
#include <conio.h>

main( )
{
    int  mat1[3][3], mat2[3][3], mat3[3][3], i, j, k, sum ;

    clrscr( ) ;

    /* get values for first matrix */
    printf ( "\nEnter values for first 3 x 3 matrix:\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
            scanf ( "%d", &mat1[i][j] ) ;
    }

    /* get values for second matrix */
    printf ( "\nEnter values for second 3 x 3 matrix:\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
```



```
            scanf ( "%d", &mat2[i][j] ) ;
    }

    /*print the first matrix as entered */
    printf ( "\nThe first 3 x 3 matrix entered by you is:\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
            printf ( "%d\t", mat1[i][j] ) ;

        printf ( "\n" ) ;
    }

    /*print the second matrix as entered */
    printf ( "\nThe second 3 x 3 matrix entered by you is:\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
            printf ( "%d\t", mat2[i][j] ) ;

        printf ( "\n" ) ;
    }

    /* calculate the product of the two matrices */
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
        {
            sum = 0 ;
            for ( k = 0 ; k <= 2 ; k++ )
                sum = sum + mat1[i][k] * mat2[k][j] ;
            mat3[i][j] = sum ;
        }
    }
    /* print the new matrix containing the product */
    printf ( "\nThe product of the above two matrices is:\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
```

```
            for ( j = 0 ; j <= 2 ; j++ )
                printf ( "%d\t", mat3[i][j] ) ;

            printf ( "\n" ) ;
        }
        printf ( "\n\nPress any key to exit..." ) ;
        getch( ) ;
    }
```

(3) Write a program to sort all the elements of a 4 x 4 matrix.

Program:

```
/* Program to sort elements of a 4 x 4 matrix */

#include <stdio.h>
#include <conio.h>

main( )
{
  int  mat[4][4], *arr, i, j, k, t ;

  clrscr( ) ;
  printf (" Enter the elements of 4 X 4 matrix:\n") ;

  for ( i = 0 ; i < 4 ; i++ )
  {
      for ( j = 0 ; j < 4 ; j++ )
          scanf ( "%d", &mat[i][j] ) ;  /* get values for the matrix */
  }
  /* print the matrix as entered */
  printf ( "\nThe 4 X 4 matrix entered by you is:\n" ) ;
  for ( i = 0 ; i < 4 ; i++ )
  {
      for ( j = 0 ; j < 4 ; j++ )
          printf ( "%d\t", mat[i][j] ) ;
```



```
        printf ( "\n" ) ;
    }

    /* sort the elements of the matrix */
    arr = mat ;  /* Base address of the matrix array */

    for ( i = 0 ; i < 15 ; i++ )  /* Number of passes */
    {
        for ( j = i + 1 ; j < 16 ; j++ )
        {
            if ( *( arr + i ) > *( arr + j ) )
            {
                t=*( arr + i ) ;
                *( arr + i ) = *( arr + j ) ;
                *( arr + j ) = t ;
            }
        }
    }

    /* print the sorted matrix */
    printf ( "\nThe sorted matrix is:\n" ) ;

    for ( i = 0 ; i < 4 ; i++ )
    {
        for ( j = 0 ; j < 4 ; j++ )
            printf ( "%d\t", mat[i][j] ) ;

        printf ( "\n" ) ;
    }

    printf ( "\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(4) Write a program to obtain the determinant value of a 5 x 5 matrix.

Program:

```
/* Determinant of a 3 x 3 matrix */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
  int  a[3][3], i, j, k, sum, p ;

  clrscr( ) ;
  printf ( "\nEnter values for a 3 x 3 matrix:\n" ) ;

  for ( i = 0 ; i <= 2 ; i++ )
  {
      for ( j = 0 ; j <= 2 ; j++ )
          scanf ( "%d", &a[i][j] ) ;  /* get values for the matrix */
  }

  /* print the matrix as entered */
  printf ( "\nEnter values for a 3 x 3 matrix:\n" ) ;
  for ( i = 0 ; i <= 2 ; i++ )
  {
      for ( j = 0 ; j <= 2 ; j++ )
          printf ( "%d\t", a[i][j] ) ;

      printf ( "\n" ) ;
  }
  /* calculate determinant */
  sum = 0 ;
  j = 1 ;
  k = 2 ;
  for ( i = 0 ; i <= 2 ; i++ )
  {
      p = pow ( -1, i ) ;  /* library function included in "math.h" */

      if ( i == 2 )
```



```
          k = 1 ;

      sum = sum + ( a[0][i] * ( a[1][j] * a[2][k] - a[2][j] * a[1][k] ) ) *
            p ;
      j = 0 ;
  }

  printf ( "\nDeterminant of the above matrix is: %d", sum ) ;

  printf ( "\n\n\n\n\nPress any key to exit..." ) ;
  getch( ) ;
}
```

Chapter 9
Puppetting On Strings

Simple strings

[A] What will be the output of the following programs:

(a)
```
main( )
{
    char  c[2] = "A" ;
    printf ( "\n%c", c[0] ) ;
    printf ( "\n%s", c ) ;
}
```

Output:

```
A
A
```

(b)
```
main( )
{
    char  s[ ] = "Get organised! learn C!!"
    printf ( "\n%s", &s[2] ) ;
    printf ( "\n%s", s ) ;
    printf ( "\n%s", &s ) ;
    printf ( "\n%c", s[2] ) ;
}
```

Output:

```
t orgainsed ! learn c !!
```

```
get organised ! learn c !!
get orgainsed ! learn c !!
t
```

(c)
```
main( )
{
    char s[ ] = "No two viruses work similarly" ;
    int i = 0 ;
    while ( s[i] != 0 )
    {
        printf ( "\n%c %c", s[i], *( s + i ) ) ;
        printf ( "\n%c %c", i[s], *( i + s ) ) ;
        i++ ;
    }
}
```

Output:

```
N  N
N  N
o  o
o  o
t  t
t  t
.....
.....
y  y
y  y
```

(d)
```
main( )
{
    char s[ ] = "Churchgate: no church no gate" ;
    char t[25] ;
    char *ss, *tt ;
    ss = s ;
    while ( *ss != '\0' )
        *ss++ = *tt++ ;
    printf ( "\n%s", t ) ;
}
```

Output:

garbage string

(e)
```
main( )
{
    char s[ ] = "Mumbadevi" ;
    char t[25] ;
    char *ss, *tt ;
    ss = s ;
    while ( *ss++ = *tt++ ) ;
        printf ( "\n%s", t ) ;
}
```

Output:

garbage string

[B] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    int arr[ ] = { `A', `B', `C', `D' } ;
    int i ;
    for ( i = 0 ; i <= 3 ; i++ )
        printf ( "\n%d", arr[i] ) ;
}
```

No Error.

(b)
```
main( )
{
    char arr[8] = "Rhombus" ;
```

```
    int i ;
    for ( i = 0 ; i <= 7 ; i++ )
        printf ( "\n%d", *arr ) ;
        arr++ ;
}
```

Error. arr cannot be incremented.

[C] Answer the following:

(a) "A" is a string while 'A' is a character.

(b) A string is terminated by a null character, which is written as ' \0 '.

(c) The array char **name[10]** can consist of a maximum of 9 characters.

(d) True or False:

The array elements are always stored in contiguous memory locations.

Answer: True

(e) Which is more appropriate for reading in a multi-word string?

gets( ) printf( ) scanf( ) puts( )

Answer:

gets( )

(f) If the string "Hackers have put us on hit list" is fed to the following set of **scanf( )** statements, what will be the contents of the array s in each case?

```
static char  s[30] ;
scanf ( "%s", &s[2] ) ;
scanf ( "%s", s ) ;
```



```
scanf ( "%s", &s ) ;
scanf ( "%c", &s[2] ) ;
scanf ( "%c", &s ) ;
```

Answer:

```
\0\0Hackers\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
have\0kers\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
put\0\0kers\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
pu \0\0kers\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
uu \0\0kers\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
```

[D] Answer the following:

(a) Observe the purpose of the functions given in Figure 9.2 and write your own functions (like say, xstrlen, xstrupr etc.) which imitate these standard library functions.

Program:

```
/* strcat( ), strcmp( ), strupr( ), strlwr( ) */

#include <stdio.h>
#include <conio.h>

main( )
{
    static char  str1[100] = "Hype and Hassels" ;
    static char  str2[100] = "Hype and Hoopla" ;
    int  l ;
    clrscr( ) ;

    xstrupr ( str1 ) ;
    printf ( "Conversion to Upper Case: %s\n", str1 ) ;

    xstrlwr ( str1 ) ;
    printf ( "Conversion to Lower Case: %s\n", str1 ) ;
```

```
    l = xstrcmp ( str1, str2 ) ;
    printf ( "Comparison Of Strings: %d\n", l ) ;

    xstrcat ( str1, str2 ) ;
    printf ( "Concatenation: %s", str1 ) ;

    printf ( "\n\n\n\n\nPress any key to exit...") ;
    getch( ) ;
}

xstrupr ( char *s )  /* conversion to upper case */
{
    while ( *s != '\0' )
    {
        if ( *s >= 97 && *s <= 122 )
            *s = *s - 32 ;
        s++ ;
    }
}

xstrlwr ( char *s )  /* conversion to lower case */
{
    while ( *s != '\0' )
    {
        if ( *s >= 65 && *s <= 90 )
            *s = *s + 32 ;
        s++ ;
    }
}

xstrcmp ( char *s, char *t )  /* comparison of strings */
{
    while ( *s == *t )
    {
        if ( *s == '\0' && *t == '\0' )
            return ( 0 ) ;
        else
        {
```



```
            s++ ;
            t++ ;
        }
    }
    return ( *s - *t ) ;
}

xstrcat ( char *s1, char *s2 )  /* concatenation of two strings */
{
    s1 = s1 + strlen ( s1 ) ;  /* go to end of 1 st string */
    while ( *s2 != '\0' )
    {
        *s1 = *s2 ;
        s1++ ;
        s2++ ;
    }
    *s1 = '\0' ;
}
```

(b) Following program uses certain functions. Observe the sample run of this program and write down the code for these user-defined functions.

```
main( )
{
    char  mess[ ] = "Do not blame me, I never voted VP" ;
    char  newmess[7] ;
    xleft ( mess, newmess, 6 ) ;
    printf ( "\n%s", newmess ) ;
    xright ( mess, newmess, 6 ) ;
    printf ( "\n%s", newmess ) ;
    xsubstr ( mess, newmess, 8, 5 ) ;
    printf ( "\n%s", newmess ) ;
}
```

The Output of this program would be...

Do not
ted VP
blame

Program:

```
/* Functions xleft( ), xright( ), xsubstr( ) */

#include <stdio.h>
#include <conio.h>

main( )
{
    static char  mess[ ] = "Do not blame me, I never voted VP" ;
    char  newmess[7] ;

    clrscr( ) ;

    xleft ( mess, newmess, 6 ) ;  /* extract 6 characters from left */
    printf ( "%s\n", newmess ) ;

    xright ( mess, newmess, 8 ) ;  /* extract 8 characters from right */
    printf ( "%s\n", newmess ) ;

    xsubstr ( mess, newmess, 8, 8 ) ;  /* extract 8 characters starting at
                                          8th character */
    printf ( "%s\n", newmess ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

xleft ( char *s, char *t, int n )  /* extract n characters from left */
{
    int i ;
    for ( i = 0 ; i < n ; i++ )
```



```
    {
        *t = *s ;
        s++ ;
        t++ ;
    }
    *t = '\0' ;
}

xright ( char *s, char *t, int n )  /* extract n characters from right */
{
    int i ;

    s = s + strlen ( s ) - n ;

    for ( i = 0 ; i < n ; i++ )
    {
        *t = *s ;
        s++ ;
        t++ ;
    }
    *t = '\0' ;
}

/* extract n characters from the string starting at a given position */
xsubstr ( char *s, char *t, int posi, int n )
{
    int i ;

    s = s + posi - 1 ;
    for ( i = 0 ; i < n ; i++ )
    {
        *t = *s ;
        s++ ;
        t++ ;
    }
    *t = '\0' ;
}
```


(c) Write a program that will print out all the rotations of a string typed into it. For example, the rotations of the word "space" are:

space paces acesp cespa espac

Program:

```
/* Print all rotations of a string */

#include <stdio.h>
#include <conio.h>

main( )
{
    char str[20] ;
    int i , j , n ;
    char *s ;

    clrscr( ) ;
    printf ( "Enter a string " ) ;
    scanf ( "%s", str ) ;

    n = strlen ( str ) ; /* length of the string entered */

    for ( i = 0 ; i < n ; i++ )
    {
        s = &str[i] ;
        for ( j = 1 ; j <= strlen(str) ; j++ )
        {
            printf ( "%c", *s ) ;
            s++ ;
            if ( *s == '\0' )
                s = str ;
        }

        printf ( "\n" ) ;
```



```
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) Write a program that replaces two or more consecutive blanks in a string by a single blank. For example, if the input is

Grim return to the planet of apes!!

the output should be

Grim return to the planet of apes!!

Program:

```
/* Replace more than one blank with a single blank */

main( )
{
    static char mess[ ] = "Grim return to the planet of apes" ;
    char newmess[50] ;
    char *s, *t ;
    int i, l ;

    clrscr( ) ;
    s = mess ;
    t = newmess ;
    l = strlen ( s ) ;

    for ( i = 0 ; i <= l-1 ; i++ )
    {
        if ( *s == ' ' ) /* check for a blank */
        {
            if ( *( s + 1 ) != ' ' )
            {
```


www.indianfun.org

```
                *t = *s ;
                t++ ;
            }
            s++ ;
        }
        else
        {
            *t = *s ;
            t++ ;
            s++ ;
        }
    }
    *t = '\0' ;

    printf ( "\nOriginal Statement: %s\n", mess ) ;
    printf ( "\nModified Statement: %s\n", newmess ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

Two dimensional array, Array of pointers to strings

[E] Answer the following:

(a) How many bytes in memory will be occupied by the following array of pointers to strings? How many bytes would be required to store the same strings, if they are stored in a two-dimensional character array?

```
char *mess[ ] = {
                    "Hammer and tongs",
                    "Tooth and nail",
                    "Spit and polish",
                    "You and C"
                };
```


www.indianfun.org

Answer:

58 bytes for storing strings using array of pointers
68 bytes for storing strings using two dimensional array

(b) Can an array of pointers to strings be used to collect strings from the keyboard? If not, why not?

Answer:

No, because when we use array of pointers it becomes necessary to initialise it at the time of declaration. And hence its storage class automatically becomes " static ".

[F] Answer the following:

(a) Write a program to count the number of `e' in the following array of pointers to strings:

```
char *s[ ] = {
                "We will teach you how to...",
                "Move a mountain",
                "Level a building",
                "Erase the past",
                "Make a million",
                "...all through C!"
            };
```

Program:

```
/* Calculate number of e's in an array of pointers to strings */

#include <stdio.h>
#include <conio.h>

main( )
{
```

www.indianfun.org

```
    static char *s[ ] = {
                            "We will teach you how to .....",
                            "Move a mountain",
                            "Level a building",
                            "Erase the past",
                            "Make a million",
                            "..... all through C !"
                        };
    char *p ;
    int i, count = 0 ;

    clrscr( ) ;

    for( i = 0 ; i < 6 ; i++ ) /* Loop for number of strings */
    {
        p = s[i] ; /* Base address of each string */

        while( *p )
        {
            printf ( "%c", *p ) ;
            if ( *p == 'e' ) /* Check for character 'e' */
                count++ ; /* count of occurence of 'e' */
            p++ ; /* next character in the string */
        }
        printf ( "\n" ) ;
    }
    printf ( "\n\nTotal number of 'e' = %d", count ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```


www.indianfun.org

(b) Develop a program that receives the month and year from the keyboard as integers and prints the calendar in the following format.

March 1995

Mon	Tue	Wed	Thu	Fri	Sat	Sun
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

Note that according to the Gregorian calendar 01/01/1900 was Monday. With this as the base the calendar should be generated.

Program:

```
/* Program to display calendar of any year */

#include <stdio.h>
#include <conio.h>
#include "goto.c"

static char *months[ ] = {
                            "January",
                            "February",
                            "March",
                            "April",
                            "May",
                            "June",
                            "July",
                            "August",
                            "September",
```

www.indianfun.org

```
                                "October",
                                "November",
                                "December"
                            };
main( )
{
    static int days[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 } ;
    long int  ndays, ldays, tydays, tdays ;
    int  d, i, m, fday, y ;
    char  ch ;

    clrscr( ) ;

    printf ( "Enter year(1900 onwards) & month (number): " ) ;
    scanf ( "%d %d", &y, &m ) ;

    ndays =  ( y - 1 ) * 365l ;
    ldays = ( y - 1 ) / 4 - ( y - 1 ) / 100 + ( y - 1 ) / 400 ;
    tdays = ndays + ldays ;

    if ( ( y % 100 == 0 && y % 400 == 0 ) || ( y % 4 == 0 &&
       y % 100 != 0 ) )
          days[1] = 29 ;
    else
          days[1] = 28 ;

    d = days[m-1] ;
    tydays = 0 ;

    for ( i = 0 ; i <= m - 2 ; i++ )
          tydays = tydays + days[i] ;

    tdays = tydays + tdays ;
    fday = tdays % 7 ;
    cal ( y, m, fday, d ) ;
}

cal ( int yr, int mo, int fd, int da )
```



```
{
    int  i, r, c ;
    char  a ;

    clrscr( ) ;
    gotorc ( 2, 25 ) ;
    printf ( "%s %d", months[mo-1], yr) ;
    gotorc ( 5, 5 ) ;
    printf ( "----------------------------------------" ) ;
    gotorc ( 6, 10 ) ;
    printf ( "Mon  Tue  Wed  Thu  Fri  Sat  Sun" ) ;
    gotorc ( 7, 5 ) ;
    printf ( "----------------------------------------" ) ;

    r = 9;
    c = 11 + 6 * fd ;
    for ( i = 1 ; i <= da ; i++ )
    {
          gotorc ( r, c ) ;
          printf ( "%d", i ) ;
          if ( c <= 41 )
                c = c + 6 ;
          else
          {
                c = 11 ;
                r = r + 1 ;
          }
    }
    gotorc ( 15, 5 ) ;
    printf ( "----------------------------------------" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(c) Modify the above program suitably so that once the calendar for a particular month and year has been displayed on the screen, then using arrow keys the user must be able to change the calendar in the following manner:

Up arrow key : Next year, same month
Down arrow key : Previous year, same month
Right arrow key : Same year, next month
Left arrow key : Same year, previous month
If the escape key is hit then the procedure should stop.

Hint: Use the **getkey( )** function discussed in Chapter 8, problem number[L](c).

Program:

```
/* Program to display calendar of any year */

#include <stdio.h>
#include <conio.h>

#include "goto.c"

static char *months[ ] = {
                          "January",
                          "February",
                          "March",
                          "April",
                          "May",
                          "June",
                          "July",
                          "August",
                          "September",
                          "October",
                          "November",
                          "December"
                          };
main( )
{
    static int days[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 } ;
    long int  ndays, ldays, tydays, tdays ;
    int  d, i, m, fday, y ;
    char  ch ;

    clrscr( ) ;

    printf ( "Enter year(1900 onwards) & month (number): " ) ;
    scanf ( "%d %d", &y, &m ) ;

    while ( 1 )
    {
        ndays =  ( y - 1 ) * 365l ;
        ldays = ( y - 1 ) / 4 - ( y - 1 ) / 100 + ( y - 1 ) / 400 ;
        tdays = ndays + ldays ; /* Total days */

        /* check for leap year */
        if ( ( y % 100 == 0 && y % 400 == 0 ) ||
           ( y % 4 == 0 && y % 100 != 0 ) )
            days[1] = 29 ;  /* days[1] = February */
        else
            days[1] = 28 ;

        d = days[m-1] ;
        tydays = 0 ;

        for ( i = 0 ; i <= m - 2 ; i++ )
            tydays = tydays + days[i] ;

        tdays = tydays + tdays ;
        fday = tdays % 7 ;
        cal ( y, m, fday, d ) ;

        ch = getkey( ) ;  /* collects keyboard input */
```

```
        switch ( ch )
        {
            case 77 :  /* right arrow - next month */
                if ( m == 12 )
                {
                    y++ ;
                    m = 1 ;
                }
                else
                    m++ ;
                break ;

            case 72 :  /* Up arrow - Next year */
                y++ ;
                continue ;

            case 75 :  /* Left Arrow - previous month */
                if ( m == 1 )
                {
                    y-- ;
                    m = 12 ;
                }
                else
                    m-- ;
                break ;

            case 80 :  /* down arrow - previous year */
                y-- ;
                continue ;

            case 1 :  /* Escape to exit */
                exit( ) ;
        }
    }
}

cal ( int yr, int mo, int fd, int da )
{
```



```
    int i, r, c ;
    char a ;

    clrscr( ) ;

    gotorc ( 2, 25 ) ;
    printf ( "%s %d ", months[mo-1], yr ) ;
    gotorc ( 5, 5 ) ;
    printf ( "-----------------------------------------------" ) ;
    gotorc ( 6, 10 ) ;
    printf ( "Mon  Tue  Wed  Thu  Fri  Sat  Sun" ) ;
    gotorc ( 7, 5 ) ;
    printf ( "-----------------------------------------------" ) ;

    r = 9 ;
    c = 11 + 6 * fd ;
    for ( i = 1 ; i <= da ; i++ )
    {
        gotorc ( r, c ) ;
        printf ( "%d", i ) ;
        if ( c <= 41 )
            c = c + 6 ;
        else
        {
            c = 11 ;
            r = r + 1 ;
        }
    }
    gotorc ( 15, 5 ) ;
    printf ( "-----------------------------------------------" ) ;
    gotorc ( 17, 11 ) ;
    printf ( "UP - Next Year        DOWN - Prev Year" ) ;
    gotorc ( 18, 11 ) ;
    printf ( "RIGHT - Next Month    LEFT - Prev Month" ) ;
    gotorc ( 20, 27 ) ;
    printf ( "Esc - Exit" ) ;
}
```

(d) Write a program to reverse the strings stored in the following array of pointers to strings:

```
char *s[ ] = {
                "To err is human...",
                "But to really mess things up...",
                "One needs to know C!!"
            };
```

Hint: Write a function **xstrrev (string)** which should reverse the contents of one string. Call this function for reversing each string stored in s.

Program:

```
/* Reverse strings stored in an array of pointers */

#include <stdio.h>
#include <conio.h>

main( )
{
   static char *s[ ] = {
                        "To ere is human...",
                        "But to really mess things up...",
                        "One needs to know C !!"
                      };
   int i ;

   clrscr( ) ;

   for( i = 0 ; i <= 2 ; i++ ) /* Three strings to be traversed */
   {
      xstrrev ( s[i] ) ;
      printf ( "%s\n", s[i] ) ;
   }
   printf ( "\n\n\n\n\nPress any key to exit..." ) ;
```



```
   getch( ) ;
}

xstrrev ( char *ss )
{
   int l, i ;
   char *tt, temp ;

   l = strlen ( ss ) ;
   tt = ss + l - 1 ;

   for ( i = 1 ; i <= l/2 ; i++ )
   {
      temp = *ss ;
      *ss = *tt ;
      *tt = temp ;
      ss++ ;
      tt-- ;
   }
}
```

Chapter 10

Structures

[A] What will be the output of the following programs:

(a)
```
main( )
{
    struct gospel
    {
        int num ;
        char mess1[50] ;
        char mess2[50] ;
    } m ;

    m.num = 1 ;
    strcpy ( m.mess1, "If all that you have is hammer" ) ;
    strcpy ( m.mess2, "Everything looks like a nail" ) ;

    /* assume that the strucure is located at address 1004 */
    printf ( "\n%u %u %u", &m.num, m.mess1, m.mess2 ) ;
}
```

Output:

Address of each structure element would be printed out

(b)
```
struct gospel
{
    int num ;
    char mess1[50] ;
    char mess2[50] ;
```

```
} m1 = { 2, "If you are driven by success",
           "make sure that it is a quality drive"
         };
main( )
{
    struct gospel m2, m3 ;
    m2 = m1 ;
    m3 = m2 ;
    printf ( "\n%d %s %s", m1.num, m2.mess1, m3.mess2 ) ;
}
```

Output:

2 If you are driven by success make sure that it is a quality drive

[B] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    struct employee
    {
        char name[25] ;
        int age ;
        float bs ;
    } ;
    struct employee e ;
    strcpy ( e.name, "Hacker" ) ;
    age = 25 ;
    printf ( "\n%s %d", e.name, age ) ;
}
```

Error. struct employee definition should be followed by a ; age cannot be accessed directly . Use e.age.

(b)
```
main( )
{
    struct
    {
        char name[25] ;
        char language[10] ;
    } ;
    struct employee e = { "Hacker", "C" } ;
    printf ( "\n%s %d", e.name, e.language ) ;
}
```

Error. %s should be used for printing e.language.

(c)
```
struct virus
{
    char signature[25] ;
    char status[20] ;
    int size ;
} v[2] = {
            "Yankee Doodle", "Deadly", 1813,
            "Dark Avenger", "Killer", 1795
         } ;
main( )
{
    int i ;
    for ( i = 0 ; i <=1 ; i++ )
        printf ( "\n%s %s", v.signature, v.status ) ;
}
```

Error. While printing signature and status, v[i].signature and v[i].status should be used.

(d)
```
struct s
{
    char ch ;
    int i ;
    float a ;
} ;
main( )
{
    struct s var = { 'C', 100, 12.55 } ;
    f ( var ) ;
```

```
    g ( &var ) ;
}
f ( struct s v )
{
    printf ( "\n%c %d %f", v -> ch, v -> i, v -> a ) ;
}
g ( struct s *v )
{
    printf ( "\n%c %d %f", v.ch, v.i, v.a ) ;
}
```

Error. In function f( ) dot operator should be used to access structure elements, while in function g( ) -> operator should be used to access structure elements.

(e)
```
struct s
{
    int i ;
    struct s *p ;
} ;
main( )
{
    struct s var1, var2 ;

    var1.i = 100 ;
    var2.i = 200 ;
    var1.p = &var2 ;
    var2.p = &var1 ;
    printf ( "\n%d %d", var1.p -> i, var2.p -> i ) ;
}
```

No Error.

[C] Answer the following:

(a) Ten floats are to be stored in memory. What would you prefer, an array or a structure?

Answer:

An array

(b) Given the statement,

```
maruti.engine.bolts = 25 ;
```

which of the following is true?

1. structure bolts is nested within structure engine
2. structure engine is nested within structure maruti
3. structure maruti is nested within structure engine
4. structure maruti is nested within structure bolts

Answer:

1. structure bolts is nested within structure engine

(c) State True or False:

1. All structure elements are stored in contiguous memory locations.

   Answer: True

2. An array should be used to store dissimilar elements, and a structure to store similar elements.

   Answer: False

3. In an array of structures, not only are all structures stored in contiguous memory locations, but the elements of individual structures are also stored in contiguous locations.

   Answer: True

(d)
```
struct time
{
    int hours ;
    int minutes ;
    int seconds ;
} t ;
struct time *tt ;
tt = &t ;
```

Looking at the above declarations, which of the following refers to **seconds** correctly:

1. tt.seconds
2. (*tt).seconds
3. time.t
4. tt -> seconds

Answer:

4. tt -> seconds

Chapter 11

Input/Output In C

Console I/O

[A] What will be the output of the following programs:

(a)
```
main( )
{
    char s[ ] = "Aw what the heck" ;
    printf ( "\n%s", s ) ;
    printf ( "\n%s", s[3] ) ;
    printf ( "\n%s", &s[3] ) ;
}
```

Output:

```
Aw what the heck
garbage
what the heck
```

(b)
```
main( )
{
    printf ( "\n%s", "Bit by bit I take a byte..." ) ;
    printf ( "\n%s", "As quiet flows the Don corleone" ) ;
}
```

Output:

```
Bit by bit I take a byte...
As quiet flows the Don corleone
```

(c)
```
char *m[ ] = {
                "Our soccer is pathetic",
                "Our politics is a scoundrels game",
                "Our economy is Ram bharose",
                "Prices have blown through the roof top",
                "C seems to be the only plus point!!"
            };
main( )
{
    int i ;
    for ( i = 0 ; i <= 4 ; i++ )
        printf ( "\n%s", m[i] ) ;
}
```

Output:

```
Our soccer is pathetic
Our politics is a scoundrels game
Our economy is Ram bharose
Prices have blown through the roof top
C seems to be the only plus point !!
```

(d)
```
char p[ ] = "The sixth sick sheikh's sixth ship is sick" ;
main( )
{
    int i = 0 ;
    while ( p[i] != '\0' )
    {
        putch ( p[i] ) ;
        i++ ;
    }
}
```

Output:

```
The sixth sick sheikh's sixth ship is sick
```


[B] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    char name[40], prof[25] ;
    puts ( "Enter name of your favourite footballer" ) ;
    scanf ( "%s", name ) ;
    puts ( "Enter your profession" ) ;
    scanf ( "%s", &prof[0] ) ;
}
```

No Error.

(b)
```
main( )
{
    char message[25] ;
    puts ( "Enter any message\n" ) ;
    gets ( "%s", message ) ;
}
```

Error. Format specifier %s cannot be used in gets( ) function.

(c)
```
main( )
{
    char key ;
    puts ( "press any key..." ) ;
    scanf ( "%c", &key ) ;
}
```

No Error. But use of getch( ) in place would have been more logical.

(d)
```
main( )
{
    char s[ ] = "Viruses at last" ;
    printf ( "\n%s", &s[0] ) ;
    printf ( "\n%s", &s[5] ) ;
```

```
}
```

No Error.

(e)
```
main( )
{
    char *mess[5] ;
    for ( i = 0 ; i < 5 ; i++ )
        scanf ( "%s", mess[i] ) ;
}
```

Error. i undefined. Moreover when we use an array of pointers to strings the array should be initialised where it is declared.

(f)
```
main( )
{
    int i ;
    float j ;
    scanf ( "%d\n%f", &i, &j ) ;
}
```

No Error. but still usage of \n is not recommended in scanf( ).

[C] Answer the following:

(a) To receive the string "We have got the guts, you get the glory!!" in an array **char str[100]** which of the following functions would you use?

1. scanf ("%s", str) ;
2. gets (str) ;
3. getche (str) ;
4. fgetchar (str) ;

Answer:

2. gets( )

(b) Which function would you use if a single key is to be received through the keyboard?

1. scanf( )
2. gets( )
3. getche( )
4. getchar( )

Answer:

3. getche( )

(c) If an integer is to be entered through the keyboard, which function would you use?

1. scanf( )
2. gets( )
3. getche( )
4. getchar( )

Answer:

1. scanf( )

(d) If a character string is to be received through the keyboard which function would work faster?

1. scanf( )
2. gets( )

Answer:

2. gets( )

(e) What is the difference between **getchar( ),fgetchar( ),getch( )** and **getche( )**?

Answer:

All receive a character from keyboard. There are minor differences in them :

getch( ) : Receives a character from keyboard without echoing it on the screen
getche( ) : Receives a character from keyboard and echos it on the screen
getchar( ) : Receives a character from keyboard, but it is necessary to hit the enter key after the character.
fgetchar( ): Same as getchar( ). getchar( ) is a macro whereas fgetchar( ) is a function.

(f) The format string of a **printf( )** function can contain:

1. characters, conversion specifications and escape sequences
2. character, integers and floats
3. strings, integers and escape sequences
4. Inverted commas, percentage sign and backslash character

Answer:

1. Characters, conversion specifications and escape sequences

(g) A field width specifier in a **printf( )** function:

1. controls the margins of the program listing
2. specifies the maximum value of a number
3. controls the size of type used to print numbers

4. specifies how many columns will be used to print the number

Answer:

4. Specifies how many columns will be used to print the number

[D] Answer the following:

(a) Write down two functions **xgets( )** and **xputs( )** which work similar to the standard library functions **gets( )** and **puts( )**.

Program:

```
/* Function xgets( ) and xputs( ) */

#include <stdio.h>
#include <conio.h>

main( )
{
   char sent[100] ;

   clrscr( ) ;

   xputs ( "Enter a sentence ... " ) ;
   xgets ( sent ) ;
   printf ( "\n\n" ) ;
   xputs ( sent ) ;
   printf ( "\n\n\n\n\nPress any key to exit..." ) ;
   getch( ) ;
}

xputs ( char *s )
{
   while(*s)
```

```
    {
        putch ( *s ) ;
        s++ ;
    }
}

xgets ( char *s )
{
    int i ;
    char ch ;

    for ( i = 0 ; i <= 98 ; i++ )
    {
        ch = getche( ) ;
        if ( ch == '\r' )
        {
            *s = '\0' ;
            break ;
        }
        if ( ch == '\b' )
        {
            printf (" \b") ;
            i-- ;
            s-- ;
        }
        else
        {
            *s = ch ;
            s++ ;
        }
    }
}
```

(b) Write down a function **getint( )** which would receive a numeric string from the keyboard, convert it to an integer number and return the integer to the calling function. A sample usage of **getint( )** is shown below:


```
main( )
{
    int a ;
    a = getint( ) ;
    printf ( "you entered %d", a ) ;
}
```

Program:

```
/* Function getint( ) */

main( )
{
    int a ;

    clrscr( ) ;

    printf ( "\nEnter a numeric string..." ) ;
    a = getint( ) ;
    printf ( "\nYou entered %d", a ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

getint( )
{
    char str[6] ;
    int i, j, k, val ;

    i = 0 ;
    while( i <= 5 )
    {
        str[i] = getche( ) ; /* string input from keyboard */
        if ( str[i] == '\r' )
        {
            str[i] = '\0' ;
            break ;
        }
```

```
        if ( str[i] == '\b' )
        {
            i-- ;
            printf (" \b") ;
        }
        else
            i++ ;
    }
    val = 0 ;
    k = 1 ;
    for ( j = i-1 ; j >= 0 ; j-- )
    {
        val = val + ( str[j] - 48 ) * k ;  /* convert to numeric value */
        k = k * 10 ;
    }
    return ( val ) ;
}
```

Disk I/O, argc, argv

[E] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    FILE *fp ;
    fp = fopen ( "TRY.C", "r" ) ;
    fclose ( fp ) ;
}
```

Error. "stdio.h" has not been included.

(b)
```
main (  )
{
    FILE fp ;
    fp == fopen ( "YOURS", "w" ) ;
    fclose ( fp ) ;
```



```
}
```

Error. fp should be declared as FILE *fp Use = in the statement to open the file.

(c)
```
#include <stdio.h>
main( )
{
    FILE *fp ;
    char c ;
    fp = fopen ( "TRY.C" ,"r") ;
    if ( fp == null )
    {
        puts ( "Cannot open file" ) ;
        exit( ) ;
    }
    while ( ( c = getc ( fp ) ) != EOF )
        putch ( c ) ;
    fclose ( fp ) ;
}
```

Error. NULL should be used instead of null.

(d)
```
#include <stdio.h>
main( )
{
    FILE *fp, *ft ;
    fp = fopen ( "TRY.C" , "r" ) ;
    fp = fopen ( "TRIAL.C", "r" ) ;
    fclose ( fp, ft ) ;
}
```

Error. Both the files are being pointed to by fp.
Separate **fclose( )** should be used for closing two different files.

(e) main (int ac, char *av[ ])

```
{
    printf ( "\n%d", ac ) ;
    printf ( "\n%s", av[0] ) ;
}
```

No Error.

[F] Answer the following:

(a) The macro FILE is defined in which of the following files:

1. stdlib.h
2. stdio.c
3. io.h
4. stdio.h

Answer:

2. stdio.h

(b) State True or False:

1. The disadvantage of High Level Disk I/O functions is that the programmer has to manage the buffers.

Answer: False

2. If a file is opened for reading it is necessary that the file must exist.

Answer: True

3. If a file opened for writing already exists its contents would be overwritten.

Answer: True

4. For opening a file in append mode it is necessary that the file should exist.

Answer: False

(c) On opening a file for reading which of the following activities are performed:

1. The disk is searched for existence of the file.
2. The file is brought into memory.
3. A pointer is set up which points to the first character in the file.
4. All the above.

Answer:

4. All the above

(d) Which of the following are the correct ways of opening a file

1.
```
#include <stdio.h>
main( )
{
   FILE *t ;
   char  filename[30] ;
   scanf ( "%s", filename ) ;
   t = fopen ( filename, "r" ) ;
   fclose ( t ) ;
}
```

2.
```
#include <stdio.h>
main( )
{
   FILE *t ;
   t = fopen ( "jazz.c", "r" ) ;
   fclose ( t ) ;
}
```

3. #include <stdio.h>
```
main ( int argc, char *argv[ ] )
{
   FILE *t ;
   t = fopen ( argv[1], "r" ) ;
   fclose ( t ) ;
}
```
4. All the above.

Answer:

4. All the above

[G] Answer the following:

(a) Write a program to count the number of words in a given text file.

Program:

```
/* Program to count total no. of words in a file */

#include <stdio.h>
#include <conio.h>

main( )
{
   /* Maximum 67 characters allowed in file name including path */
   char str[80], filename[67], ch ;
   int blanks, i, len ;
   FILE *fs ;

   clrscr( ) ;
   puts ( "Enter filename: " ) ;
   gets ( filename ) ;
   fs = fopen ( filename, "r" ) ;  /* read only mode */
   if ( fs == NULL )
   {
```



```
      puts ( "Cannot open file" ) ;
      printf ( "\n\n\n\n\nPress any key to exit..." ) ;
      getch( ) ;

      exit( ) ;
   }

   /* count number of spaces in the file */
   blanks = 0 ;
   while ( fgets ( str, 100, fs ) != NULL )
   {
      len = strlen ( str ) - 2 ;
      for ( i = 0 ; i < len ; i++ )
      {
         if ( str[i] == ' ' && str[i+1] != ' ' )
            blanks++ ;
      }
      blanks = blanks + 1 ;
   }
   fclose ( fs ) ;
   printf ( "Number of words ... %d ", blanks ) ;

   printf ( "\n\n\n\n\nPress any key to exit..." ) ;
   getch( ) ;
}
```

(b) Write a program to store every character typed at the keyboard into a file. The procedure should come to an end as soon as the character ~ is hit from the keyboard.

Program:

```
/* Store every character typed at the keyboard into a file */

#include <stdio.h>
#include <conio.h>
```

```
main( )
{
    char  filename[67] ;
    FILE *fp ;
    char  ch ;

    clrscr( ) ;
    puts ( "Enter file name " ) ;
    gets ( filename ) ;

    fp = fopen ( filename, "w" ) ; /* write mode */
    if ( fp == NULL )
    {
        puts ( "Can not open a file" ) ;

        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    else
    {
        printf ( "\nType some sentences..." ) ;
        printf ( "\nType a '~' to stop.\n" ) ;
    }

    while ( ( ch = getche( ) ) != '~' ) /* input from keyboard */
    {
        if ( ch == '\r' )
        {
            printf ( "\n" ) ;
            putc ( '\n', fp ) ;
            continue ;
        }
        else
            putc ( ch, fp ) ;
    }
```



```
    puts ( "\nFile successfully created & saved" ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

    fclose ( fp ) ;
}
```

(c) Write a program to remove all blank lines from a file.

Program:

```
/* Program to remove all the blank lines from a file */

#include <stdio.h>
#include <conio.h>
#define PRINTABLE 1
#define NOT_PRINTABLE 0

main ( int  argc, char  *argv[ ] )
{
    FILE  *fs, *ft ;
    char  ch, str[100] ;
    int  i = 0, line ;

    clrscr( ) ;

    if ( argc != 3 )
    {
        puts ( "Insufficient  No. of parameters ... " ) ;
        exit( ) ;
    }
    fs = fopen ( argv[1], "r" ) ;  /* source file - read only mode */
    if ( fs == NULL )
    {
        puts ( "Cannot open source file ... " ) ;
```

```
        printf ( "\n\n\n\n\nPress any key to exit:.." ) ;
        getch( ) ;

        exit( ) ;
    }
    ft = fopen ( argv[2], "w" ) ; /* target file - write mode */
    if ( ft == NULL )
    {
        puts ( "Cannot open target file ... " ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        fclose ( fs ) ;
        exit( ) ;
    }
    line = NOT_PRINTABLE ;
    while ( ( ch = getc ( fs ) ) != EOF )
    {
        if ( ch != '\n' )
        {
            str[i] = ch ;
            i++ ;
            if ( ch != ' ' && ch != '\t' )
            line = PRINTABLE ;
            continue ;
        }
        else
        {
            str[i] = '\0' ;
            if ( line == PRINTABLE )
            {
                printf ( "%s\n", str ) ;
                fprintf ( ft, "%s\n", str ) ;
                line = NOT_PRINTABLE ;
            }
            i = 0 ;
        }
    }
```



```
    fclose ( fs ) ;
    fclose ( ft ) ;

    printf ( "\nMission Accomplished. " ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) Write a program to remove all comments from a C program file.

Program:

```
/* Remove comments from a C program */

#include <stdio.h>
#include <conio.h>

FILE *fs, *ft ;

main ( )
{
    /* Maximum 67 characters allowed in file name including path */
    char source[67], target[67] ;
    int c, d ;

    puts ( "Enter C file name ..." ) ;
    gets ( source ) ;
    fs = fopen ( source, "r" ) ;
    if ( fs == NULL )
    {
        puts ( "Cannot open source file : " ) ;

        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
```

```
        exit( ) ;
    }
    puts ( "Enter target file name : " ) ;
    gets ( target ) ;
    ft = fopen ( target, "w" ) ;
    if ( ft == NULL )
    {
        puts ( "Cannot open target file ... " ) ;
        fclose ( fs ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
        exit( ) ;
    }

    while ( ( c = getc ( fs ) ) != EOF )
    {
        if ( c == '/' )
        {
            if ( ( d = getc ( fs ) ) == '*' )
                incomment( ) ;
            else
            {
                fprintf ( ft, "%c", c ) ;
                fprintf ( ft, "%c", d ) ;
            }
        }
        else
            fprintf ( ft, "%c", c ) ;
    }
    fclose ( fs ) ;
    fclose ( ft ) ;

    printf ( "\nMission Accomplished. ") ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```



```
incomment( )
{
    int c, d ;

    c = getc ( fs ) ;
    d = getc ( fs ) ;
    while ( c != '*' || d != '/' )
    {
        c = d ;
        d = getc (fs ) ;
    }
}
```

(e) Write a program to display the contents of a text file on the screen. Make following provisions:

Display the contents inside a box drawn with opposite corner co-ordinates being (1, 0) and (23, 79). Display the name of the file whose contents are being displayed, and the page numbers in the zeroth row. The moment one screenful of file has been displayed, flash a message `Press any key...' in 24th row. When a key is hit, the next page's contents should be displayed, and so on till the end of file.

Program:

```
/* Display the contents of a file */

#include <stdio.h>
#include <conio.h>
#include "goto.c"

main ( int argc, char *argv[ ] )
{
    FILE *fp ;
    char ch ;
    int pg = 1, r = 2, c = 1
```

```
clrscr( ) ;

if ( argc != 2 )
{
     puts ( "Insufficient  No. of parameters ... " ) ;
     printf ( "\n\n\n\n\nPress any key to exit..." ) ;
     getch( ) ;

     exit( ) ;
}
fp = fopen ( argv[1], "r" ) ;
if ( fp == NULL )
{
     puts ( "Cannot open source file ... " ) ;
     printf ( "\n\n\n\n\nPress any key to exit..." ) ;
     getch( ) ;

     exit( ) ;
}

printf ( "gft" ) ;
cls( ) ;

drawbox ( 1, 0, 22, 79 ) ;
gotorc ( 0, 0 ) ;
printf ( "Filename : %s", argv[1] ) ;
gotorc ( 0, 65 ) ;
printf ( "Page :- %d", pg ) ;

while ( ( ch = getc(fp) ) != EOF )
{
    if ( ch != '\n' )
    {
        gotorc ( r, c ) ;
        printf ( "%c", ch ) ;
        c++ ;
    }
```



```
        else
        {
            r++ ;
            c = 1 ;
            if ( r > 20 )
            {
                pg++ ;
                r = 2 ;
                gotorc ( 23, 0 ) ;
                puts ( "Press any key ... " ) ;
                getch( ) ;
                cls( ) ;
                drawbox ( 1, 0, 22, 79 ) ;
                gotorc ( 0, 0 ) ;
                printf ( "Filename : %s", argv[1] ) ;
                gotorc ( 0, 65 ) ;
                printf ( "Page : %d", pg ) ;
            }
        }
    }
    getch( ) ;
    fclose ( fp ) ;
}

drawbox ( int  a, int  b, int  i, int  j )
{

    int  x, k ;
    gotorc ( a, b ) ;
    printf ( "%c", 201 ) ;
    gotorc ( i, b ) ;
    printf ( "%c", 200 ) ;
    gotorc ( i, j ) ;
    printf ( "%c", 188 ) ;
    gotorc ( a, j ) ;
    printf ( "%c", 187 ) ;

    for ( x = a + 1 ; x <= i - 1 ; x++ )
```



```
    {
        gotorc ( x, b ) ;
        printf ( "%c", 186 ) ;
        gotorc ( x, j ) ;
        printf ( "%c", 186 ) ;
    }
    for ( k = b + 1 ; k <= j - 1 ; k++ )
    {
        gotorc ( a, k ) ;
        printf ( "%c", 205 ) ;
        gotorc ( i, k ) ;
        printf ( "%c", 205 ) ;
    }
}
```

(f) Write a program to encrypt/decrypt a file using:

(1) An offset cipher: In an offset cipher each character from the source file is offset with a fixed value and then written to the target file.

For example, if character read from the source file is 'A', then convert this into a new character by offsetting 'A' by a fixed value, say 128, and then writing the new character to the target file.

Program:

```
/* Encrypt / Decrypt a file using offset cipher */

#include <stdio.h>
#include <conio.h>

FILE *fs ,*ft ;

clrscr( ) ;
```



```
main ( int argc, char *argv[ ] )
{
  if ( argc != 4 )
  {
      puts ( "Improper command .. Correct usage CH11GF1
              <source file> <target file> <C/D>" ) ;
      printf ( "\n\n\n\n\nPress any key to exit..." ) ;
      getch( ) ;

      exit( ) ;
  }
  fs = fopen ( argv[1], "r" ) ;
  if ( fs == NULL )
  {
      printf ( "Cannot open source file" ) ;
      puts ( argv[1] ) ;
      printf ( "\n\n\n\n\nPress any key to exit..." ) ;
      getch( ) ;

      exit( ) ;
  }
  ft = fopen ( argv[2], "w" ) ;
  if ( ft == NULL )
  {
      puts ( "cannot open target file " ) ;
      puts ( argv[2] ) ;
      fclose ( fs ) ;
      printf ( "\n\n\n\n\nPress any key to exit..." ) ;
      getch( ) ;

      exit( ) ;
  }
  if ( *argv[3] == 'c' || *argv[3] == 'C' )
      code( ) ;
  else
      decode( ) ;
  fclose ( fs ) ;
  fclose ( ft ) ;
```

```
    printf ( "\nMission Accomplished. " ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

code( )
{
    int ch ;

    while ( ( ch = getc ( fs ) ) != EOF )
    {
        ch = ch + 128 ; /* encrypt - offset each character by 128 */
        putc ( ch, ft ) ;
    }
}

decode( )
{
    int ch ;

    while ( ( ch = getc ( fs ) ) != EOF )
    {
        ch = ch - 128 ; /* decrypt - offset each character by -128 */
        putc ( ch, ft ) ;
    }
}
```

(2) A substitution cipher: In this each character read from the source file is substituted by a corresponding predetermined character and this character is written to the target file.

For example, if character 'A' is read from the source file, and if we have decided that every 'A' is to be substituted by '!', then a '!' would be written to the target file in place

of every 'A' Similarly, every 'B' would be substituted by '5' and so on.

Program:

```
/* Encrypt / Decrypt a file using substitution cipher */

#include <stdio.h>
#include <conio.h>

FILE *fs, *ft ;

main ( int argc, char *argv[ ] )
{
    if ( argc != 4 )
    {
        puts ( "Improper command ... " ) ;
        puts ( "Correct Usage : CH11GF2 <source file> <target
                file> C/D" ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    fs = fopen ( argv[1], "r" ) ; /* source file - read only mode */
    if ( fs == NULL )
    {
        puts ( "Cannot open source file ... " ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    ft = fopen ( argv[2], "w" ) ; /* target file - write mode */
    if ( ft == NULL )
    {
        puts ( "Cannot open target file ... " ) ;
```

```
        fclose ( fs ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
        exit( ) ;
    }
    if ( *argv[3] == 'c' || *argv[3] == 'C' )
        code( ) ;
    else
    {
        if ( *argv[3] == 'd' || *argv[3] == 'D' )
            decode( ) ;
        else
        {
            fclose ( fs ) ;
            fclose ( ft ) ;
            puts ( "Improper command ... " ) ;
            printf ( "\n\n\n\n\nPress any key to exit..." ) ;
            getch( ) ;

            exit( ) ;
        }
    }
    fclose ( fs ) ;
    fclose ( ft ) ;
}

code( )
{
    char ch ;
    int i = 0 ;
    static char arr1[97] = "IOP{}asdfghjkl;'ASDFGHJKL:zxcvbn
                            m,./ZXCVBNM<>?`1234567890-=\~!@#
                            $%^&*( )_+|qwertyuiop[ ]QWERTYU" ;

    static char arr2[97] = "`1234567890-=\~!@#$%^&*( )_+|qw
                            ertyuiop[]QWERTYUIOP{}asdfghjkl;'A
                            SDFGHJKL:zxcvbnm,./ZXCVBNM<>?" ;
    arr2[93] = '\\' ;
```



```
    arr1[93] = '\\' ;
    arr2[94] = '\"' ;
    arr1[94] = '\"' ;
    arr2[95] = '\n' ;
    arr1[95] = '\n' ;
    arr1[96] = '\t';
    arr2[96] = '\t' ;
    while ( ( ch = getc ( fs ) ) != EOF )
    {
        for ( i = 0 ; i <= 96 ; i++ )
        {
            if ( ch == arr1[i] )
                break ;
        }
        putc ( arr2[i], ft ) ;
    }
}

decode( )
{
    char ch ;
    int i = 0 ;
    static char arr1[97] = " IOP{}asdfghjkl;'ASDFGHJKL:zxcvbn
                            m,./ZXCVBNM<>?`1234567890-=~!@#
                            $%^&*( )_+|qwertyuiop[ ]QWERTYU" ;

    static char arr2[97] = " `1234567890-=~!@#$%^&*( )_+|qw
                            ertyuiop[ ]QWERTYUIOP{}asdfghjkl;'A
                            SDFGHJKL:zxcvbnm,./ZXCVBNM<>?" ;
    arr2[93] = '\\' ;
    arr1[93] = '\\' ;
    arr2[94] = '\"' ;
    arr1[94] = '\"' ;
    arr2[95] = '\n' ;
    arr1[95] = '\n' ;
    arr1[96] = '\t';
    arr2[96] = '\t' ;
    while ( ( ch = getc ( fs ) ) != EOF )
```

```
        {
            for ( i = 0 ; i <= 96 ; i++ )
            {
                if ( ch == arr2[i] )
                    break ;
            }
            putc ( arr1[i], ft ) ;
        }
    }
```

String I/O, Formatted disk I/O, Text and binary modes

[H] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    FILE *fp ;
    char str[80] ;
    fp = fopen ( "TRY.C", "r" ) ;
    while ( fgets ( str, 80, fp ) != EOF )
        fputs ( str ) ;
    fclose ( fp ) ;
}
```

Error. stdio.h should be included fputs( ) needs two arguments NULL should be used with fgets( ) to check end of file.

(b)
```
main( )
{
    FILE *fp ;
    char str[80] ;

    fp = fopen ( "MYPROG.C", "r" ) ;
    while ( fgets ( str, fp ) != EOF )
    printf ( "\n%s" , str ) ;
```



```
    fclose ( fp ) ;
}
```

Error. stdio.h should be included fgets( ) needs three arguments NULL should be used with fgets( ) to check end of file.

(c)
```
main( )
{
    FILE *fp ;
    char name[25] ;
    int age ;

    fp = fopen ( "YOURS", "r" ) ;
    while ( fscanf ( fp, "%s %d", name, &age ) != NULL )
    fclose ( fp ) ;
}
```

Error. stdio.h should be included EOF should be used with fscanf( ) to check end of file.

[I] Answer the following:

(a) Is it necessary that a file created in text mode must always be opened in text mode for subsequent operations?

Answer:

Yes

(b) State True or False:

A file opened in binary mode and read using fgetc( ) would report the same number of characters in the file as reported by DOS's DIR command.

Answer: True

(c) While using the statement,

```
fp = fopen ( "myfile.c", "r" ) ;
```

what happens if,

- myfile.c does not exist on the disk
- myfile.c exists on the disk

Answer:

If myfile.c does not exist on the disk it returns a NULL
If myfile.c exists on the disk then the file is loaded into memory and a file pointer is set up which points to the first character in the file

(d) What is the purpose of the library function **fflush( )**?

Answer:

Clear the specified buffer fflush (stdin) clears the keyboard buffer

(e) While using the statement,

```
fp = fopen ( "myfile.c", "wb" ) ;
```

what happens if,

- myfile.c does not exist on the disk.
- myfile.c exists on the disk

Answer:

If file does not exist on disk then a new file is created
If file exists on disk then the file is destroyed and a new file is created

(f) A floating point array contains percentage marks obtained by students in an examination. To store these marks in a file "marks.c", in which mode would you open the file and why?

Answer:

Open the file in binary mode since in binary mode a float occupies only four bytes when stored in a file unlike the text mode file in which number of bytes occupied by a float depends upon the length of the float

[J] Answer the following:

(a) Given below are two programs written in dBASE. The first one is poorly written. Contents-wise both the programs are same, but the second one has been properly indented (the IF's & ENDIF's, the DOWHILE's & ENDDO's, the DO CASE's & ENDCASE's have been properly lined up).

Program One

```
* A sample dbase program
INPUT "Enter any number" TO N
DO WHILE ( .T. )
IF ( N > 30 )
A = 25
ELSE
A = 30
ENDIF
INPUT "Enter any number" TO CODE
DO CASE
CASE CODE = 1
B = 100
CASE CODE = 2
B = 200
```

```
CASE CODE = 3
B = 300
CASE CODE = 4
B = 400
ENDCASE
ENDDO
```

Program Two

```
* A sample dbase program
INPUT "Enter any number" TO N
DO WHILE ( .T. )
    IF ( N > 30 )
        A = 25
    ELSE
        A = 30
    ENDIF
    INPUT "Enter any number" TO CODE
    DO CASE
        CASE CODE = 1
            B = 100
        CASE CODE = 2
            B = 200
        CASE CODE = 3
            B = 300
        CASE CODE = 4
            B = 400
    ENDCASE
ENDDO
```

Write a program which would receive as input the first program from a source file and Output the properly indented second progran into a target file.

Program:

```
/* Program to indent a dbase program */
```



```
#include <stdio.h>
#include <conio.h>

main( )
{
    static char spaces[ ] = "    " ;
    FILE *fs, *ft ;
    /* Maximum 67 characters allowable in file name including path */
    char source[67], target[67], str[100], b[150], a[100] ;
    int spc = 0, dw, ifs, elses, endif, enddo, docase, cases, endcase ;

    clrscr( ) ;

    puts ( "Enter source file name ..." ) ;
    gets ( source ) ;

    fs = fopen ( source, "r" ) ; /* source file - read only mode */
    if ( fs == NULL )
    {
        puts ( "Cannot open source file ... " ) ;
        puts ( source ) ;

        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
        exit( ) ;
    }

    puts ( "Enter target file name ..." ) ;
    gets ( target ) ;

    ft = fopen ( target, "w" ) ; /* target file - write mode */
    if ( ft == NULL )
    {
        puts ( "Cannot open target file ... " ) ;
        puts ( target ) ;
        fclose ( fs ) ;
```

```
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
        exit( ) ;
}

while ( fgets ( str, 100, fs ) != NULL )
{
    trim ( str, a ) ;
    dw = strncmp ( a, "DO WHILE", 8 ) ;
    if ( dw == 0 )
    {
        xstrdup ( b, spaces, spc ) ;
        strcat( b, a ) ;
        fprintf ( ft, "%s", b ) ;
        spc++ ;
        *b = '\0' ;
        continue ;
    }
    ifs = strncmp ( a, "IF", 2 ) ;
    if ( ifs == 0 )
    {
        xstrdup ( b, spaces, spc ) ;
        strcat( b, a ) ;
        fprintf ( ft, "%s", b ) ;
        spc++ ;
        *b = '\0' ;
        continue ;
    }
    elses = strncmp ( a, "ELSE", 4 ) ;
    if ( elses == 0 )
    {
        spc-- ;
        xstrdup ( b, spaces, spc ) ;
        strcat ( b, a ) ;
        fprintf ( ft,"%s", b ) ;
        spc++ ;
        *b = '\0' ;
        continue ;
```



```
    }
    endif = strncmp ( a, "ENDIF", 5 ) ;
    if ( endif == 0 )
    {
        spc-- ;
        xstrdup ( b, spaces, spc ) ;
        strcat ( b, a ) ;
        fprintf ( ft,"%s", b ) ;
        *b = '\0' ;
        continue ;
    }
    enddo = strncmp ( a, "ENDDO", 5 ) ;
    if ( enddo == 0 )
    {
        spc-- ;
        xstrdup ( b, spaces, spc ) ;
        strcat( b, a ) ;
        fprintf ( ft, "%s", b ) ;
        *b = '\0' ;
        continue ;
    }
    docase = strncmp ( a, "DO CASE", 6 ) ;
    if ( docase == 0 )
    {
        xstrdup ( b, spaces, spc ) ;
        strcat( b, a ) ;
        fprintf ( ft, "%s", b ) ;
        spc += 2 ;
        *b = '\0' ;
        continue ;
    }
    cases = strncmp ( a, "CASE", 4 ) ;
    if ( cases == 0 )
    {
        spc-- ;
        xstrdup ( b, spaces, spc ) ;
        strcat ( b, a ) ;
        fprintf ( ft, "%s", b ) ;
```

```
            spc++ ;
            *b = '\0' ;
            continue ;
        }
        endcase = strncmp ( a, "ENDCASE", 7 ) ;
        if ( endcase == 0 )
        {
            spc =spc - 2 ;
            xstrdup ( b, spaces, spc ) ;
            strcat ( b, a ) ;
            fprintf ( ft, "%s", b ) ;
            *b = '\0' ;
            continue ;
        }
        xstrdup ( b, spaces, spc ) ;
        strcat( b, a ) ;
        fprintf ( ft, "%s", b ) ;
        *b = '\0' ;
    }
}

xstrdup ( char *t, char *s, int n )
{

    int i ;

    for ( i = 1 ; i <= n ; i++ )
        strcat ( t, s ) ;
}

trim ( char *s, char *t )
{
    while ( *s == ' ' || *s == '\t' )
        s++ ;
    strcpy ( t, s ) ;
}
```


(b) Write a program, which prints out only those lines from a file, which are containing more then 80 character. Also print out the line numbers of these lines.

Program:

```
/* Print out lines containing more than 80 characters */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *fs ;
    char source[67], str[200] ;
    int lineno = 1 ;

    clrscr( ) ;

    puts ( "Enter source file name ..." ) ;
    gets ( source ) ;

    fs = fopen ( source, "r" ) ;
    if ( fs == NULL )
    {
        puts ( "Cannot open source file ... " ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }

    while ( fgets ( str, 200, fs ) != NULL )
    {
        if ( strlen ( str ) > 80 )
            printf ( "%d : %s\n", lineno, str ) ;
```

```
        lineno = lineno + 1 ;
    }
    fclose ( fs ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

}
```

(c) Write a program that will concatenate two files: that is, append the contents of one file at the end of another and write the result into a third file. You must be able to execute the command at DOS prompt as follows:

C>CONCAT source1.txt source2.txt target.txt

Program:

```
/* Program to concatenate two files */

#include <stdio.h>
#include <conio.h>

main ( int argc, char *argv[ ] )
{
    FILE *fs1, *fs2, *ft ;
    char ch ;

    clrscr( ) ;

    if ( argc != 4 )
    {
        puts ( " Improper argument " ) ;
        puts ( " Correct usage CH11JC < source1 > < source2 > <
                target >" ) ;
        exit( ) ;
    }
    fs1 = fopen ( argv[1], "r" ) ;
    if ( fs1 == NULL )
    {
        printf ( "Cannot open file %s", argv[1] ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    fs2 = fopen ( argv[2], "r" ) ;
    if ( fs2 == NULL )
    {
        printf ( "Cannot open file %s", argv[2] ) ;
        fclose ( fs1 ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    ft = fopen ( argv[3], "w" ) ;
    if ( ft == NULL )
    {
        printf ( "Cannot open file %s", argv[3] ) ;
        fclose ( fs1 ) ;
        fclose ( fs2 ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }

    while ( ( ch = getc ( fs1 ) ) != EOF )
        putc ( ch, ft ) ;
    fclose ( fs1 ) ;

    while ( ( ch = getc ( fs2 ) ) != EOF )
        putc ( ch, ft ) ;
```

```
    fclose ( fs2 ) ;

    fclose ( ft ) ;
    printf ( "\n\nTwo files have been combined together. " ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) A file contains a C program, but by mistake a novice has typed it out in capitals. It's a long program, therefore you cannot afford to have it retyped. Write a utility, which reads this program, converts it to an appropriate small-case program, and writes it to a target file. Use functions **fgets( )** and **fputs( ).**

Program:

```
/* Convert capital lettered file into small case letter file */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *fs, *ft ;
    char str[80], source[67], target[67] ;

    puts ( "Enter source file name" ) ;
    gets ( source ) ;
    fs = fopen ( source, "r" ) ;
    if ( fs == NULL )
    {
        puts ( "Cannot open source file" ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    puts ( "Enter target file name " ) ;
    gets ( target ) ;
    ft = fopen ( target, "w" ) ;
    if ( ft == NULL )
    {
        puts ( "Cannot open target file" ) ;
        fclose ( fs ) ;

        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }

    while ( fgets ( str, 80, fs ) != NULL )
    {
        strlwr ( str ) ;
        fputs ( str, ft ) ;
    }
    fclose ( fs ) ;
    fclose ( ft ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

}
```

Record I/O, Low level disk I/O

[K] Point out the errors, if any, in the following programs:

(a)
```
main( )
{
    FILE *fp ;
    char names[20] ;
    int i ;
```



```
    fp = fopen ( "students.c", "wb" ) ;
    for ( i = 0 ; i <= 10 ; i++ )
    {
        puts ( "\nEnter name " ) ;
        gets ( name ) ;
        fwrite ( name, size of ( name ), 1, fp ) ;
    }
    close ( fp ) ;
}
```

Error. <stdio.h> should be included sizeof should be one word
fclose( ) should be used to close the file.

(b)
```
main( )
{
    FILE *fp ;
    char name[20] = "Ajay" ;
    int i ;
    fp = fopen ( "students.c", "r" ) ;
    for ( i = 0 ; i <= 10 ; i++ )
        fwrite ( name, sizeof ( name ), 1, fp ) ;
    close ( fp ) ;
}
```

Error. <stdio.h> should be included fclose( ) should be used to close the file.

(c)
```
#include "fcntl.h"
main( )
{
    int fp ;
    fp = open ( "pr22.c" , "r" ) ;
    if ( fp == -1 )
        puts ( "cannot open file" ) ;
    else
        close ( fp ) ;
```



```
}
```

Error. O_RDONLY should be used in place of "r" while opening the file.

(d)
```
main( )
{
    int fp ;
    fp = fopen ( "students.c", READ | BINARY ) ;
    if ( fp == -1 )
        puts ( "cannot open file" ) ;
    else
        close ( fp ) ;
}
```

Error. "stat.h" should be included instead of fopen( ) it should be open O_RDONLY and O_BINARY should be used.

[L] Answer the following:

(a) In the file "STUD.DAT" there are 100 records with the following structure:

```
struct students
{
    int rollno ;
    float permarks ;
} ;
```

In another file "STUDENTS.DAT" there are 100 records with the following structure:

```
struct studinfo
{
    int rollno , centreno ;
    char branch[10];
    float per ;
```

```
};
```

However, in this file in all the 100 records no entry exists in the field **per**. Write a program to copy the percentage marks from the file "STUD.DAT" into the file "STUDENTS.DAT". See to it that data other than the percentage marks in "STUDENTS.DAT" remains undisturbed. Note that the roll numbers present in the two files are not necessarily in the same order.

Program:

```
/* Program to copy a field from a data file to another data file */

#include <stdio.h>
#include <conio.h>

struct stud
{
    int roll ;
    float permarks ;
};

struct studinfo
{
    int rollno ;
    int centreno ;
    char branch[10] ;
    float per ;
};

main( )
{
    struct stud s ;
    struct studinfo ss ;
    FILE *fs , *ft ;
    int slen = sizeof ( struct studinfo ) ;
```



```
    fs = fopen ( "STUD.DAT", "rb" ) ; /* read only in binary mode */
    if ( fs == NULL )
    {
        puts ( "Cannot open file : STUD.DAT" ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    ft = fopen ( "STUDENTS.DAT", "rb+" ) ;/* read, write & modify in
                                              binary mode */
    if ( ft == NULL )
    {
        puts ( "Cannot open file : STUDENTS.DAT" ) ;
        fclose ( fs ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }

    while ( fread ( &s, sizeof(s), 1, fs ) == 1 )
    {
        fseek ( ft, 0, SEEK_SET ) ;
        while ( fread ( &ss, slen, 1, ft ) == 1 )
        {
            if ( s.roll == ss.rollno )
            {
                ss.per = s.permarks ;
                fseek ( ft, -slen, SEEK_CUR ) ;
                fwrite ( &ss, slen, 1, ft ) ;
                break ;
            }
        }
    }
    fclose ( fs ) ;
    fclose ( ft ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
```

```
    getch( ) ;
}
```

(b) There are 100 records present in a file with the following structure:

```
struct
{
    char itemcode[6], itemname[20] ;
    int qty ;
} ;
```

Write a program to read these records, arrange them in ascending order and write them in to a target file.

Program:

```
/* Arrange records in a file in ascending order */

#include <stdio.h>
#include <conio.h>

main( )
{
    struct item
    {
        char itemcode[6] ;
        char itemname[20] ;
        int qty ;
    } ;

    struct item it[100], s, temp ;
    int j, k ;
    FILE *fs, *ft ;
    int n ;
    fs = fopen ( "ITEM.DAT", "rb" ) ; /* read only in binary mode */
```



```
    if ( fs == NULL )
    {
        puts ( "Cannot open file : ITEM.DAT" ) ;

        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    ft = fopen ( "ASCITEM.DAT", "wb" ) ; /* write in binary mode */
    if ( ft == NULL )
    {
        puts ( "Cannot open file : ASCITEM.DAT" ) ;
        fclose ( fs ) ;
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit( ) ;
    }
    n = 0 ;
    while ( fread ( &s, sizeof(s), 1, fs ) == 1 )
    {
        it[n] = s ;
        n++ ;
    }
    for ( j = 0 ; j < n ; j++ )
    {
        for ( k = j+1 ; k < n ; k++ )
        {
            if ( strcmp ( it[j].itemcode, it[k].itemcode ) > 0 )
            {
                temp = it[j] ;
                it[j] = it[k] ;
                it[k] = temp ;
            }
        }
    }
```

```
    for ( j = 0 ; j < n ; j++ )
        fwrite ( &it[j], sizeof(s), 1, ft ) ;

    fclose ( fs ) ;
    fclose ( ft ) ;

    printf ( "\n\nRecords arranged in ascending order " ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```


Chapter 12
Fundamental Computer Concepts

[A] Pick up the correct alternative for each of the following:

(a) Which of the following are not housed on the mother board

(1) Memory chips
(2) Support chips
(3) Disk Drive
(4) Microprocessor

Answer:

(3) Disk Drive

(b) Which of the following doesn't belong to Intel 8086 family of microprocessor

(1) 8086
(2) 8088
(3) 80286
(4) 68000

Answer:

(4) 68000

(c) Which of the following is a math co-processor

(1) 8088
(2) 8087
(3) 8086
(4) 80286

Answer:

(2) 8087

(d) Which microprocessor is used in PC/AT

(1) 8088
(2) 8086
(3) 80286
(4) 8085

Answer:

(3) 80286

(e) Which of the following task is not done by the microprocessor

(1) Arithmetic operations
(2) Logical comparison operations
(3) Store data in CPU registers
(4) Write information to disk

Answer:

(4) Write information to disk

(f) A computer which has 8086 as its CPU is called

(1) 8/16-bit computer
(2) 16/32-bit computer
(3) 16-bit computer
(4) 8-bit computer

Answer:

(3) 16-bit computer

(g) Typically, a PC/AT has got

(1) 2 Floppy disk drives
(2) 2 Hard disk drives
(3) 1 Floppy disk drive, 1 Hard disk drive
(4) None of the above

Answer:

(3) 1 Floppy disk drive, 1 hard disk drive

(h) Programs stored in which of the following memories cannot be erased

(1) RAM
(2) ROM
(3) Cache memory
(4) Virtual memory

Answer:

(2) ROM

(i) Which out of the following is the fastest microprocessor

(1) 8086
(2) 80286
(3) 80386
(4) 8088

Answer:

(3) 80386

(j) Which of the following microprocessor cannot work in Real as well as in Protected mode

(1) 8088
(2) 80286
(3) 80386
(4) 80486

Answer:

(1) 8088

(k) A 32-bit microprocessor can at a time handle

(1) 32 bits
(2) 32 bytes
(3) 32 kilobytes
(4) 32 megabytes

Answer:

(1) 32 bits

(l) Pick the odd one out

(1) PC
(2) PC/XT
(3) Minicomputer
(4) PC/AT

Answer:

(3) Minicomputer

[B] State True or false:

(a) Disk drives are plugged into expansion slots.

Answer: False

(b) Programs stored in RAM are permanent and cannot be modified.

Answer: False

(c) To a 16-bit microprocessor a 32-bit data bus can be connected.

Answer: False

(d) New data cannot be stored in RAM during program execution.

Answer: False

(e) A data bus is a set of wires connecting the microprocessor and memory through which data flows.

Answer: True

(f) Every PC needs to have a math co-processor.

Answer: False

(g) If while executing a program the power goes off, all the instructions and data in the memory will be lost.

Answer: True

(h) A PC/XT uses a 80186 microprocessor whereas a PC/AT uses a 80286 microprocessor.

Answer: False

Chapter 13

Disk Basics

[A] Pick up the correct alternative for each of the following:

(a) DOS identifies the way a disk has been formatted by

(1) Format ID
(2) Media descriptor
(3) Number of tracks on the disk
(4) Number of sectors on the disk

Answer:

(2) Media descriptor

(b) The boot sector of a disk contains

(1) Disk bootstrap program
(2) Bootstrap loader program
(3) Directory entries
(4) Information about when the disk was formatted

Answer:

(1) Disk bootstrap program

(c) Length of each directory entry is

(1) 34 bytes
(2) 32 bytes
(3) 36 bytes

(4) 8 bytes

Answer:

(2) 32 bytes

(d) A cluster represents

(1) A group of tracks
(2) A group of sectors
(3) Total number of tracks present on the disk
(4) Total number of sectors present on the disk

Answer:

(2) A group of sectors

(e) The date field in the directory entry is

(1) 2 bytes long
(2) 8 bytes long
(3) 6 bytes long
(4) None of the above

Answer:

(1) 2 bytes long

(f) Maximum length of a volume label entry is

(1) 8 bytes
(2) 3 bytes
(3) 11 bytes
(4) None of the above

Answer:

(3) 11 bytes

(g) The status of the files IO.SYS and MSDOS.SYS is usually

(1) Read/Write
(2) System and Hidden
(3) Read only
(4) System, Read only and Hidden

Answer:

(4) System, Read only and Hidden

(h) Where a particular file begins in the data space is indicated by

(1) Starting cluster number
(2) FAT entry of the file
(3) Boot parameters of the file
(4) None of the above

Answer:

(1) Starting cluster number

(i) The entry of starting cluster of a file is present in

(1) Boot parameters
(2) Directory
(3) File allocation table
(4) Data space

Answer:

(2) Directory

(j) On hard disk side 0, track 0, sector 1 contains

(1) Disk bootstrap
(2) Directory
(3) File allocation table
(4) Partition table and master boot program

Answer:

(4) Partition table and master boot program

(k) The size of a partition table on hard disk is

(1) 64 bytes
(2) 32 bytes
(3) 16 bytes
(4) 64 bits

Answer:

(4) 64 bytes

(l) Partition table contains information about

(1) Where the different partitions of the disk begin
(2) Where the different partitions of the disk end
(3) Which is the bootable partition
(4) All the above

Answer:

(4) All the above

(m) While booting from the hard disk the disk bootstrap is loaded into memory by

(1) Master boot program
(2) Bootstrap loader program
(3) COMMAND.COM
(4) IO.SYS

Answer:

(1) Master boot program

(n) If a virus is to infect your disk, which is the possible where it would lodge itself

(1) Boot sector
(2) File allocation table
(3) Directory
(4) Data space

Answer:

(1) Boot sector

[B] State whether the following statements are True or False:

(a) All diskettes can be made bootable.

Answer: True

(b) A bootable diskette is one which contains operating files.

Answer: True

(c) The place where IO.SYS and MSDOS.SYS files can be present the disk is fixed.

Answer: True

(d) BIOS refers to a sector by its logical number whereas refers to it by side number, track number and sector number.

Answer: False

(e) All disk formats are upwardly compatible.

Answer: True

(f) Innermost track on the floppy disk is track number 0.

Answer: False

(g) Like a hard disk, a floppy disk can also be partitioned.

Answer: False

(h) On a hard disk more than one operating system can be .

Answer: True

(i) A hard disk must always be partitioned.

Answer: False

(j) Number of sectors occupied by directory and file allocation changes from format to format.

Answer: True

[C] Write a program to read the boot parameters from boot sector of a floppy disk and display them on the screen

Hint: Use the standard library function **_bios_disk( )** if you are using Quick C and the function **absread( )** if you are Turbo C.

Program:

```
/* Display boot parameters of floppy */

#include <dos.h>
#include <stdio.h>
#include <conio.h>

main( )
{
    struct boot
    {
        unsigned char code[3] ;
        unsigned char system_id[8] ;
        int bytes_per_sec ;
        char sec_per_clus ;
        int res_sec ;
        char fat_copies ;
        int root_dir_entry ;
        unsigned int no_sects ;
        unsigned char format_id ;
        int sec_per_fat ;
        int sec_per_trk ;
        int no_sides ;
        int no_sp_res_sect ;
        unsigned char rest_code[482] ;
    } ;
    struct boot b ;
    char temp[4] ;
    int val, drive ;

    val = absread ( 0, 1, 0, &b ) ;
    if ( val == -1 )
    {
        printf ( "Disk read Error...bad sector\n" ) ;
        printf ( "\n\n\nPress any key to exit..." ) ;
        getch( ) ;

        exit(1) ;
    }

    clrscr( ) ;

    printf ( "System id %s\n", b.system_id ) ;
    printf ( "Bytes per sector %d\n", b.bytes_per_sec ) ;
    printf ( "Sectors per cluster %d\n", b.sec_per_clus ) ;
    printf ( "Reserved sectors %d\n", b.res_sec ) ;
    printf ( "FAT copies %d\n", b.fat_copies ) ;
    printf ( "Root directory entries %d\n", b.root_dir_entry ) ;
    printf ( "No. of sectors on disk %u\n", b.no_sects ) ;
    printf ( "Format id %X\n", b.format_id ) ;
    printf ( "Sectors per FAT %d\n", b.sec_per_fat ) ;
    printf ( "Sectors per track %d\n", b.sec_per_trk ) ;
    printf ( "No. of sides %d\n", b.no_sides ) ;
    printf ( "No. of reserved sectors %d\n", b.no_sp_res_sect ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
```

}

Chapter 14

Operating System Fundamentals

[A] Pick up the correct alternative for each of the following:

(a) Which of the following is NOT true about an Operating System

(1) It acts as a manager
(2) It acts as a command processor
(3) It acts as a controller
(4) It acts as a compiler

Answer:

(4) It acts as a compiler

(b) Which of the following is NOT true about an Operating System

(1) Single user
(2) Batch processing
(3) Multi user
(4) Multi tasking

Answer:

(2) Batch Processing

(c) MS-DOS is a

(1) Single user operating system
(2) Multi user operating system
(3) Multi tasking operating system
(4) None of the above

Answer:

(1) Single user operting system

(d) O S/2 is a

(1) Single user operating system
(2) Multi user operating system
(3) Multi tasking operating system
(4) None of the above

Answer:

(3) Multi tasking operating system

(e) Who was the originator of MS-DOS

(1) Tim Patterson
(2) Gary Kildall
(3) Dennis Ritchie
(4) Ken Thompson

Answer:

(1) Tim Patterson

(f) Which of the following operating systems are supported IBM Compatible microcomputers

(1) Unix
(2) Xenix
(3) MS-DOS

(4) All the above

Answer:

(4) All the above

(g) At a time how many operating system can be at work on computer

(1) Only one
(2) Two
(3) Three
(4) Four

Answer:

(1) Only one

(h) Pick the odd one out

(1) IO.SYS
(2) MSDOS.SYS
(3) ROM-BIOS
(4) COMMAND.COM

Answer:

(3) ROM-BIOS

(i) Which of the following are components of IO.SYS

(1) API and Disk BIOS
(2) Disk BIOS and SYSINIT
(3) SYSINIT and API
(4) SYSINIT and Kernel

Answer:

(2) Disk BIOS and SYSINIT

(j) Which of the following belongs to COMMAND.COM

(1) Shell Portion
(2) Kernel Portion
(3) Batch file Portion
(4) Resident Portion

Answer:

(1) Shell Portion

(k) Which of the following is responsible for displaying DOS prompt

(1) Resident portion of COMMAND.COM
(2) Transient portion of COMMAND.COM
(3) SYSINIT module of IO.SYS
(4) AUTOEXEC.BAT

Answer:

(2) Transient portion of COMMAND.COM

(l) Which of the following remains in memory temporarily

(1) Resident portion of COMMAND.COM
(2) Transient portion of COMMAND.COM
(3) API
(4) Disk BIOS

Answer:

(2) Transient portion of COMMAND.COM

(m) Which of the following is responsible for loading of portion of COMMAND.COM

(1) Resident portion of COMMAND.COM
(2) MSDOS.SYS

(3) SYSINIT module of IO.SYS
(4) Bootstrap loader

Answer:

(1) Resident portion of COMMAND.COM

(n) Non-standard equipment attached to the computer is checked initialized by

(1) ROM startup software
(2) ROM extension software
(3) ROM BIOS software
(4) ROM Basic software

Answer:

(2) ROM extension software

(o) Which of the following allows us to interact with the

(1) ROM startup software
(2) ROM extension software
(3) ROM BIOS software
(4) ROM Basic software

Answer:

(3) ROM BIOS software

(p) Bootstrap loader program is a program belonging to:

(1) ROM startup software
(2) ROM extension software
(3) ROM BIOS software
(4) ROM Basic software

Answer:

(1) ROM startup software

[B] State True or False:

(a) Transient portion of COMMAND.COM is loaded on the top resident portion of COMMAND.COM.

Answer: False

(b) B ootstrap loader program and disk bootstrap program are one and the same.

Answer: False

(c) Interrupt Vector Table contains addresses of ROM-BIOS routines.

Answer: True

(d) The file AUTOEXEC.BAT is executed by transient portion of COMMAND.COM.

Answer: False

(e) Commands like DIR, COPY, DEL are interpreted by the program stored in the file IO.SYS.

Answer: False

(f) These days the microcomputers do not contain ROM Basic software.

Answer: True

(g) Integrity check of the ROM routines is done every time we switch on the computer.

Answer: True

(h) ROM-BIOS software cannot be modified by a user.

Answer: True

(i) ROM software is chip-based whereas DOS software is disk-based.

Answer: True

[C] The memory size of your computer is stored at location 0x413 and 0x414. Write a program to pickup this memory size display it on the screen. What can you conclude if this value 't turn out to be a standard value?

Program:

```
/* Program to display memory size */

#include <stdio.h>
#include <conio.h>

main( )
{
    int far *f = 0x413 ;
    printf ( "Size of memory = %d", *f ) ;
    getch( ) ;
}

/* If the size doesn't turn out to be a standard value like 256 or 640 kb
   then there is a possibility that a virus is active in the memory */
```

Chapter 15

VDU Basics

[A] State True or False

(a) High refresh rates cause the screen to flicker contributing to eye strain.

Answer: False

(b) The microprocessor does not have the ability to send signals necessary to produce the images on the screen

Answer

(c) Text or graphics on the screen are built up rrom tiny dots called picture elements or 'pixels'.

Answer: True

(d) The flat screen results in reduced glare and a higher quality, more accurate image.

Answer: True

(e) Interlaced monitors offer better and stable displays as compared with non-interlaced monitors.

Answer: False

(f) The graphics card in a modern PC can be connected either to the PCI slot or to the AGP slot.

Answer: True

(g) In text mode for each character on screen there are two bytes in video memory, one containing the ASCII value of the character and the other containing its color.

Answer: True

(h) The amount of memory that is required for representing a character on screen in text mode and a pixel in graphics mode is always same.

Answer: False

(i) VGA's video memory is organized in four planes and each plane provides one bit of data for each pixel.

Answer: True

(j) In VGA since each DAC register is 18 bits long, a pixel can have any of the 2,62,144 values (2^{18}).

Answer: True

(k) In SVGA graphics mode, each DAC register is 24 bits long.

Answer: True

(l) While accessing video memory under DOS, A block is used for the text mode and B block is used for the Graphics mode.

Answer: False

[B] Pick up the correct alternative for each of the following:

(a) Which of the following would display a message on the screen fastest

(1) Using printf( )function
(2) Using puts( ) function
(3) Using ROM-BIOS services
(4) Directly writing to VDU memory

Answer:

(4) Directly writing to VDU memory

(b) A multiple-frequency monitor is also called

(1) Multisync monitor
(2) Multiscan monitor
(3) Asynchronous monitor
(4) All of the above

Answer:

(4) All of the above

(c) The monitors that sweeps the screen in lines from top to bottom one line after the other in one pass are known as

(1) Non-interlaced monitors
(2) Interlaced monitors
(3) FST monitors
(4) All the above

Answer:

(1) Non-interlaced monitors

(d) The first graphics card, introduced in August of 1981 by IBM, was known as

(2) Monochrome Display Adapter (MDA)
(3) Hercules Graphics Card
(4) Color Graphics Adapter (CGA)

(5) Enhanced Graphics Adapter(EGA)

Answer:

(1) Monochrome Display Adapter (MDA)

(e) In the text mode each character displayed on the screen occupies _____ bytes in VDU memory

(1) 2
(2) 4
(3) 6
(4) 1

Answer:

(1) 2

(f) The total number of DAC registers in VGA card are

(1) 64
(2) 128
(3) 256
(4) 512

Answer:

(3) 256

[C] Answer the following:

(a) What do you mean by refresh rate? Is it true that higher the refresh rate better is the image on the screen?

Answer:

The display adapter circuitry repeatedly reads information from VDU memory and places it on the screen, making the

images displayed on the screen clear and steady. This process is called as 'Refreshing the screen', and the rate at which the display adapter refreshes the screen is called as "refresh rate".

(b) What is the purpose of the display adapter? Where is it present in the computer?

Answer:

To pick the contents of VDU memory and place them on the screen. It is plugged into one of the expansion slots.

(c) Why is it that in text mode there is only one font available, whereas in graphics mode characters can be displayed in a variety of fonts?

Answer:

In text mode each character is generated by a program called character generator that is hard wired into the display adapter and hence can generate only one type of character. In graphics mode each character is a memory map of corresponding values in the VDU memory. Since we can put different values in VDU memory we can generate different fonts.

(d) What color byte would you use if a message is to be displayed with brown background and yellow coloured characters?

Answer:

01101110

(e) When we change over from one mode to another do the screen's current contents remain intact?

Answer:

No

[D] Answer the following:

(a) In the following program the displaymenu( ) function is supposed to write a given menu on a given VDU page by directly writing it in VDU memory. You are required to write the function displaymenu( ). Make it as general as you can.

```
char *filemenu[ ] = {
                        "Rename file",
                        "Copy file",
                        "Delete file",
                        "Display file"
                    };

char *dirmenu[ ] = {
                        "Make directory",
                        "Change directory",
                        "Remove directory"
                        "List directory"
                    };
main( )
{
    int row = 5, col = 20, vdupage, num ;

    num = 4 ; /* number of menu items */
    vdupage = 0 ;
    displaymenu ( filemenu, row, col, vdupage, num ) ;
    vdupage = 1 ;
    displaymenu ( dirmenu, row, col, vdupage, num ) ;
}
```

Program:

www.indianfun.org


```
/* Write menus on different VDU pages */

#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include "goto.c"

static char *filemenu[ ] = {
                        "Rename file",
                        "Copy file",
                        "Delete file"
                    };

static char *dirmenu[ ] = {
                        "Make directory",
                        "Change directory",
                        "Remove directory",
                    };

main( )
{
    int row = 5, col = 20 , vdupage ;

    clrscr( ) ;
    vdupage = 0 ;
    displaymenu ( filemenu, row, col, 3, vdupage ) ;
    gotoxy ( 20 , 12 ) ;
    puts ( "Press any key to continue.." ) ;
    getch( ) ;

    vdupage = 1 ;
    displaymenu ( dirmenu, row, col, 3, vdupage ) ;

    writestring ( "Press any key to exit...",12,20,1) ;

    getch( ) ;
}
```

www.indianfun.org

```
displaymenu ( char **menu, int r, int c, int count, int page )
{
    int i ;
    for ( i = 0 ; i < count ; i++ )
    {
        writestring ( menu[i], r, c, page ) ;
        r++ ;
    }
}

writestring ( char *s, int r, int c, int page )
{
    while( *s )
    {
        writechar ( *s, 112, r, c, page ) ;
        s++ ;
        c++ ;
    }
}

writechar ( char ch, int attr, int r, int c, int page )
{
    char far *vdumem = 0xb8000000 ;
    char far *v ;

    /* Set active display page */
    union REGS i, o ;
    i.h.ah = 5 ; /* service number */
    i.h.al = page ;
    int86 ( 0x10, &i, &o ) ; /* interrupt number */

    /* 80 x 2 = 160 bytes for each row */
    /* even numbered columns - 1 st byte of each location */
    /* each vdu page occupies 4 Kb = 4096 bytes of memory */

    v = vdumem + r * 160 + c * 2 + page * 4096 ;
    *v = ch ;
    v++ ;
```



```
    /* attribute = binary 01110000 = 112 produces black character
       on white background */
    *v = attr ;
}
```

(b) Write a general purpose function writestring( ) which will display a message on the screen by writing it directly into VDU memory. The function should be capable of displaying the message in the attribute, which is sent to it.

Program:

```
/* writestring( ) function */

#include <stdio.h>
#include <dos.h>
#include <conio.h>

main( )
{
    char message[100] ;
    int row, col, attr ;

    clrscr( ) ;

    puts ( "Enter message, row & col no., and attribute:\n" ) ;
    gets ( message ) ; /* this will accept spaces also in the message */
    scanf ( "%d %d %d", &row, &col, &attr ) ;
    writestring ( message, attr, row, col ) ;

    gotoxy ( 1 , 24 ) ;
    puts ( "Press any key to exit..." ) ;
    getch( ) ;
}

writestring ( char *s, int attr, int r, int c )
{
```

```
    while( *s )
    {
        writechar ( *s, attr, r, c ) ;
        s++ ;
        c++ ;
    }
}

writechar ( char ch, int attr, int r, int c )
{
    char far *vdumem = 0xb8000000 ;
    char far *v ;

    v = vdumem + r * 160 + c * 2 ;
    *v = ch ;
    v++ ;
    *v = attr ;
}
```

(c) Write a program which continuously keeps changing the capital letters present on the screen into small case letters and small case letters present on the screen into capitals. You are not allowed to use printf( ), putchar( ), puts( ) or putch( ).

Program:

```
/* Keep changing capital letters on the screen to small letters & vice
   versa */

#include <stdio.h>
#include <conio.h>

main( )
{
    char far *scr = 0xb8000000 ;
    int i ;
```



```
    /* continue loop until a key is hit */
    while ( ! kbhit( ) )
    {
        for ( i = 0 ; i <= 3999 ; i++ )
        /* number of bytes = 80 x 25 x 2 = 4000 */
        {
            /* change contents of location from uppercase to
               lowercase */
            if ( *( scr + i ) >= 65 && *( scr + i ) <= 90 )
                *(scr + i) = *( scr + i ) + 32 ;
            else
            {
                /* change contents of location from lowercase to
                   uppercase */
                if ( *( scr + i ) >= 97 && *( scr + i ) <= 122 )
                    *(scr + i) = *( scr + i ) - 32 ;
            }
        }
    }
}
```


www.indianfun.org

Chapter 16
Keyboard Basics

[A] Answer the following:

(a) What are the actions of following key combinations?

(1) Ctrl-Alt-Del

Answer:

Computer boots up.

(2) Ctrl-C

Answer:

Generates the "break"character to terminates the execution of program.

(3) Shift-PrtSc

Answer:

The contents present on the screen are sent to the printer.

(4) Ctrl-Num Lock

Answer:

Stop the scrolling.

(b) What is the difference between shift keys and toggle keys?

Answer:

Shift keys : These keys change the shift state and hence change the meaning of whatever key is pressed along with it. Left shift, Right shift, Ctrl and Alt are shift keys.

Toggle keys : Toggle keys are activated with a single keystroke and remain active until released by another keystroke. They also affect the keyboard's shift state. Caps lock and Num lock are toggle keys.

(c) What are the contents of the two-byte sequence generated on hitting a key?

Answer:

The first byte contains the Ascii value of the key hit and the second byte contains the Scan code of the key hit.

(d) What do you mean by `Typematic Rate' and `Typematic Delay"? Can these be changed through a program?

Answer:

If a key is pressed for more than 0.5 seconds then the auto repeat action begins and the key is repeated about 10 times in a second. This 0.5 seconds duration is called as typematic delay and the rate at which the key is repeated is known as typematic rate.

(e) Is it true that two different scan codes are generated, one on pressing a key and another on releasing it?

Answer:

Yes

(f) How do you type out graphics characters on the screen?

Answer:

- Keep Alt key depressed.
- Hit number corresponding to ascii value of graphic character from numeric keypad.
- Release the Alt key.

(g) If the caps lock is right now ON and you hit the key combination shift A, how would this combination be interpreted by the computer using bytes 0x417 and 0x418?

Answer:

Since caps lock is on, bit 6 of byte 0x417 is set to 1. When you hit shift A, firstly bit 1 or 0 of 0x417 is put on. Then the scan code of the key a is received. Since the caps lock is on and shift is pressed we get a small case a on the screen.

(h) If a program is being executed and you hit 2-3 keys, how would these keys be tackled? Where would these keys be stored? Maximum how many keys can be stored?

Answer:

These keys would not immediately appear on the screen since the microprocessor is busy executing the program. But these keys would get stored in the keyboard buffer and would appear on the screen when the program execution terminates. Maximum of 15 keys can be stored in the keyboard buffer. If we try to store more than 15 keys the speaker beeps.

[B] Answer the following:

(a) Write a program, which puts on a capital lock. Make use of the coding scheme given earlier for bytes 0x417 and 0x418.

Program:

```
/* Program to put caps lock ON */

main( )
{
    char far *status = 0x417 ;
    /* bit no. 6 of the byte should be set to 1 */
    *status = *status | 64 ;
}
```

(b) Write a function, which allows the user to hit a key from the keyboard. If an ordinary key is hit the function should return the ascii code of the key hit, whereas if a special key is hit, the function should return the scan code of the key.

Program:

```
/* Function getkeyhit( ) which returns asciicode / scancode of key hit */

#include <stdio.h>
#include <conio.h>

main( )
{
    int ch ;

    clrscr( ) ;

    puts ( "\nHit any key.. " ) ;
```


www.indianfun.org


```
    ch = getkey ( ) ;

    printf ( "\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

getkey( ) /* function to collect a key hit */
{
    int ascii, scan ;

    while ( !kbhit( ) ) ; /* continue loop until a key is hit */
        ascii = getch( ) ;

    if ( ascii == 0 )
    {
        scan = getch( ) ;
        printf ( "You hit a special key of scancode %d", scan ) ;
    }
    else
    {
        printf ( "You hit a ordinary key of ascii value %d", ascii ) ;
    }
}
```

www.indianfun.org

Chapter 17

Interaction With Hardware Through C

[A] Pick up the correct alternative for each of the following

(a) To interact with the hardware which of the following is a reasonably reliable as well as a reasonably fast procedure

(1) Using high level language functions
(2) Using ROM-BIOS functions
(3) Directly programming the hardware
(4) None of the above

Answer:

(2) Using ROM-BIOS function

(b) Each address present in Interrupt Vector Table is

(1) 4 bytes long
(2) 2 bytes long
(3) 1 bytes long
(4) 8 bytes long

Answer:

(1) 4 bytes long

(c) Pick the odd one out

(1) Scratch-pad register
(2) Segment registers
(3) Flags register
(4) Interrupt registers

Answer:

(4) Interrupt register

(d) Whether a particular operation is successfully carried out by DOS or not is indicated by the value stored in

(1) Flags register
(2) Ordinary variables
(3) Segment register
(4) Offset register

Answer:

(1) Flags register

(e) In a 8088 microprocessor each CPU register is

(1) 12 bits long
(2) 16 bits long
(3) 8 bits long
(4) 8 bytes long

Answer:

(2) 16 bits long

(f) The process of storing the contents of CPU register into the memory when an interrupt occurs, is called

(1) Pushing values on the stack
(2) Popping values off the stack

(3) Setting up a stack pointer
(4) Setting up an instruction pointer

Answer:

(1) Pushing values on the stack

(g) A ROM-BIOS routine makes use of the segment registers ES and DS. To call this routine which function would you use

(1) int86( )
(2) int86x( )
(3) intdos( )
(4) intdosx( )

Answer:

(2) int86x( )

(h) While calling a ROM-BIOS/DOS routine, the service number should always be placed in

(1) AH register
(2) AL register
(3) AX register
(4) None of the above

Answer:

(1) AH register

[B] What will be the output of the following programs:

(a)
```
main( )
{
    union
    {
        int i[2];
```

```
        long l ;
    } u ;

    u.i[0] = 0x50 ;
    u.i[1] = 0x45 ;
    printf ( "\n%lu", u.l ) ;
}
```

Output:

4522064

(b)
```
main( ) .
{
    union a
    {
        int i1 ;
        int i2 ;
        char s[4] ;
    } ;
    union a m ;
    strcpy ( a.s, "AAA" ) ;
    printf ( "\n%s", a.s ) ;
    printf ( "\n%u %u", a.i1, a.i2 ) ;
}
```

Output:

AAA 16705 16705

[C] State whether the following statements are True or False:

(a) If ROM-BIOS functions are used in a program, they would increase the size of the executable file, the way standard library functions do.

Answer: False

(b) ROM-BIOS and DOS functions do not have names.

Answer: True

(c) The routine which gets executed when an interrupt occurs is called an Interrupt Service Routine.

Answer: True

(d) An interrupt can occur through hardware as well as software.

Answer: True

(e) All ROM-BIOS services return a value indicating success or failure in AX register.

Answer: False

(f) A union variable occupies as much data as what a corresponding structure variable would occupy, elements of both remaining same.

Answer: False

(g) Union provides us a way to look at the same memory locations in more than one way, which a structure cannot.

Answer: True

[D] Attempt the following:

NOTE: For most of the problems given below you would be required to invoke ROM-BIOS routines. The interrupt number, service number and other requirements of these routines are given in Appendix G and H.

(a) Write a general purpose function **goto_rc( )** to place the cursor at any given position on the screen. Test this by displaying the message "Rat a tat tat" at 10_th_ row, 20_th_ column on the screen.

Program:

```
/* Function goto_rc( ) to place the cursor at any given position */

#include <stdio.h>
#include <dos.h>
#include <conio.h>

main( )
{
    clrscr( ) ;

    goto_rc ( 10, 20 ) ;
    printf ( "Rat A Tat Tat" ) ;
    goto_rc ( 15 , 20 ) ;
    printf ( "Press any key to exit..." ) ;
    getch( ) ;
}

goto_rc ( int  x, int  y )
{
    union REGS  i, o ;
    i.h.ah = 2 ;
    i.h.dl = y ;
    i.h.dh = x ;
    i.h.bh = 0 ;
    int86 ( 0x10, &i, &o ) ;
}
```

(b) Write a general purpose function **size( )** which when called would change the size of the cursor in text mode. The function should be intelligent enough to hide the cursor when called upon to do so.

Program:

```
/* Change the size of cursor */

#include <stdio.h>
#include <dos.h>
#include <conio.h>

main( )
{
    int  z, x ;

    clrscr( ) ;

    puts ( "Select starting & ending dimensions of cursor
            ( between 0 & 7 ).\n" ) ;
    puts ( "To hide cursor enter ending value = 0.\n " ) ;
    scanf ( "%d %d", &z, &x ) ;

    shape ( z, x ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

shape ( int  z, int  x )
{
    union REGS  i, o ;
    i.h.ah = 1 ;  /* service number */
    i.h.ch = z ;  /* height */
    i.h.cl = x ;  /* width */
    int86 ( 16, &i, &o ) ;  /* interrupt number */
}
```

(c) Write a program which would pick up all digits currently present on screen and replace them with smiling face.

Program:

```
/* Replace digits on screen by smiling face */

#include <stdio.h>
#include <dos.h>
#include <conio.h>

main( )
{
    char far  *scr = 0xb8000000 ;
    int  i ;

    for ( i = 0 ; i <= 3999 ; i++ )
        if ( *( scr + i ) >= 48 && *( scr + i ) <= 57 )
            *(scr + i) = 1 ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) Write a function **cls( )** which would wipe the contents of the screen and place the cursor on top corner of the screen when called.

Program:

```
/* Program to clear the screen */

#include <dos.h>

main( )
{
    cls( ) ;
}

cls( )
{
    union REGS  i, o ;

    i.h.ah = 6 ;  /* service number */
    i.h.al = 0 ;
    i.h.ch = 0 ;
    i.h.cl = 0 ;
    i.h.dh = 24 ;
    i.h.dl = 79 ;
    i.h.bh = 7 ;
    int86 ( 0x10, &i, &i ) ;  /* interrupt number - video services */
}
```


(e) Implement the following procedure in a program:

Fill the entire screen with an alphabet 'A', then create a blank window of 16 rows by 30 columns on the screen. In this window display first 16 ascii values and their corresponding characters. As soon as the window is full, wipe out the window and fill it with the next 16 characters. Continue the process till you have displayed the entire ascii table.

Program:

```
/* Dispaly Ascii table in a window */

#include <stdio.h>
#include <dos.h>
#include "goto.c"

main( )
{
    int  j, i, r, c ;
    char far  *scr = (char far *) 0xB8000000 ;  /* VDU memory address */

    cls( ) ;
```

```
    for ( i = 0 ; i < 4000 ; i+=2 )
        *( scr + i ) = 'A' ; /* fill the screen with "A" */

    clearw( 2, 24, 21, 53 ) ; /* create a window */

    gotorc ( 2, 32 ) ;
    printf ( "Ascii Table" ) ;

    r = 4 ;
    c = 30 ;

    for ( i = 0 ; i <= 255 ; i++ )
    {
        gotorc ( r, c ) ;
        printf ( "%d      %c", i, i ) ; /* print ASCII table */
        r++ ;
        if ( r > 19 )
        {
            gotorc ( 21, 30 ) ;
            puts ( "Press Any Key ..." ) ;
            getch( ) ;
            r = 4 ;
            clearw ( 3, 24, 21, 53 ) ;
        }
    }
}

clearw ( int r1, int c1, int r2, int c2 )
{
    union REGS i, o ;
    i.h.ah = 6 ; /* service number */
    i.h.al = 0 ;
    i.h.ch = r1 ; /* window co-ordinates */
    i.h.cl = c1 ;
    i.h.dh = r2 ;
    i.h.dl = c2 ;
    i.h.bh = 7 ;
    int86 ( 16, &i, &o ) ; /* interrupt number */
```



```
}
```

(f) While developing a package it is often required to display different messages with different attributes. Write a program to display the message "Tutu" in all possible attributes. Alongside each message, mention the attribute value, which has been used to display the message.

Program:

```
/* Display TuTu in various colour combinations */

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include "goto.c"

main( )
{
    char str[7] ;
    int at, ir, c, j, i, r ;
    static int bgattr[ ] = { 0, 16, 32, 48, 64, 80, 96, 112 } ;
    static char *bgcol[ ] = { "Black", "Blue", "Green", "Cyan",
                              "Red",  "M'nta","Brown", "White" } ;

    cls( ) ;
    drawcolbox1 ( 0, 0, 20, 79, 112 ) ;/* draw box in inverse video */

    for ( i = 0 ; i <= 7 ; i++ )
    {
        r = 3 * i + 2 ;
        writestring ( bgcol[i], r, 0, 112 ) ;
    }
    i = 0 ;
    for ( r = 1 ; r <= 22 ; r += 3 )
    {
```

```
            j = 0 ;
            for ( c = 6 ; c <= 76 ; c += 5 )
            {
                drawcolbox1 ( r, c, r+1, c+3, bgattr[i] ) ; /* draw 4 x 4 box */
                if ( j == i ) /* background & character colour same */
                    j++ ;
                at = bgattr[i] + j ;
                writestring ( "TuTu", r, c, at ) ;
                itoa ( at, str, 10 ) ; /* convert int to char */
                writestring ( str, r + 1, c + 1, at ) ;
                j++ ;
            }
            i++ ;
        }
        gotorc ( 23, 0 ) ;
        getch ( ) ;
}

drawcolbox1 ( int tr, int tc, int br, int bc, char attr )
{
    int r, c ;
    for ( r = tr ; r <= br ; r++ )
    {
        for ( c = tc ; c <= bc ; c++ )
            writechar ( ' ', r, c, attr ) ;
    }
}

writestring ( char *ba, int r, int c, char attr )
{
    while ( *ba )
    {
        writechar ( *ba, r, c, attr ) ;
        ba++ ;
        c++ ;
    }
}
```



```
writechar ( char ch, int r, int c, char attr )
{
    char far *v ;
    v = 0xb8000000 + r * 160 + c * 2 ;
    *v = ch ;
    v++ ;
    *v = attr ;
}
```

(g) Write a program to draw a line from coordinates (10, 10) to (150, 150). What is the limitation of your program?

Program:

```
/* Draw line from 10,10 to 150,150 */

#include <dos.h>

main( )
{
    int x, y ;
    setmode ( 6 ) ;
    for ( x = 10 ; x <= 150 ; x++ )
        writedot ( x, x ) ;
    getch( ) ;
}

setmode ( int m )
{
    union REGS i, o ;
    i.h.ah = 0 ;
    i.h.al = m ;
    int86 ( 16, &i, &o ) ;
}

writedot ( int xx, int yy )
```



```
{
    union REGS i, o ;
    i.h.ah = 12 ;
    i.h.al = 1 ;
    i.x.cx = xx ;
    i.h.dl = yy ;
    int86 ( 16, &i, &o ) ;
}

/* Limitation: Can draw lines inclined at 45 degrees only. */
```

(h) Write a function **freehand( )** which when called allows you to draw freehand drawing. **freehand( )** must allow drawing interactively. Four arrow keys should allow drawing in four direction, whereas PgUp, Pgdn, Home and End keys should allow drawing diagonally.

Program:

```
/* Freehand drawing */

#include <stdio.h>
#include "goto.c"
#include <dos.h>

main( )
{
    char ch ;
    int  x = 10, y = 10, p = 1 ;

    cls( ) ;

    setmode ( 6 ) ;
    while ( 1 )
    {
        writedot ( x, y, 1 ) ;
```



```
    ch = getkey( ) ;  /* keyboard input */
    switch ( ch )
    {
        /* the number keys from numeric pad are to be used
            in numlock off mode only */
        case 75 :  /* left arrow or 4 */
            x-- ;
            break ;

        case 77 :  /* right arrow or 6 */
            x++ ;
            break ;

        case 72 :  /* down arrow or 2 */
            y-- ;
            break;

        case 80 :  /* up arrow or 8 */
            y++ ;
            break ;

        case 73 :  /* page dn or 3 */
            x++ ;
            y-- ;
            break ;

        case 81 :  /* page up or 9 */
            x++ ;
            y++ ;
            break ;

        case 71 :  /* end or 1 */
            x-- ;
            y-- ;
            break ;

        case 79 :  /* home or 7 */
            x-- ;
```


www.indianfun.org

```
                y++ ;
                break ;

            case 1 :
                exit( ) ;

        }
    }
}
setmode ( int m )
{
    union REGS i, o ;
    i.h.ah = 0 ;
    i.h.al = m ;
    int86 ( 16, &i, &o ) ;
}

writedot ( int xx, int yy, int p )
{
    union REGS i, o ;
    i.h.ah = 12 ;
    i.h.al = p ;
    i.x.cx = xx ;
    i.h.dl = yy ;
    int86 ( 16, &i, &o ) ;
}
```


Chapter 18

Operations On Bits

[A] Answer the following:

(a) Using DOS service number 67 (refer Appendix H for details of this service) write a program to hide a given file from a given directory.

Program:

```
/* Program to hide a file using DOS service no 67 */

# include <stdio.h>
# include <conio.h>
# include <dos.h>

main( )
{
    union REGS i, o ;
    char filename[67] ;

    puts ( "Enter filename to hide:\n" ) ;
    gets ( filename ) ;

    i.h.ah = 67 ; /* service number - 0x43 */
    i.x.dx = filename ;
    i.h.al = 1 ;
    i.x.cx = 2 ; /* file attributes - bit number 1 set for hidden */
    intdos ( &i, &o ) ;
```

```
    if ( o.x.cflag != 0 )  /* carry flag set indicating an Error */
    {
        puts ( "File not found" ) ;
        printf ( "\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
    }
    else
    {
        printf ( "\n\nFile  %s  is now hidden.", filename ) ;
        printf ( "\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;
    }
}
```

(b) Using DOS service number 67 write a program to display the current attributes of any file in a given directory.

Program:

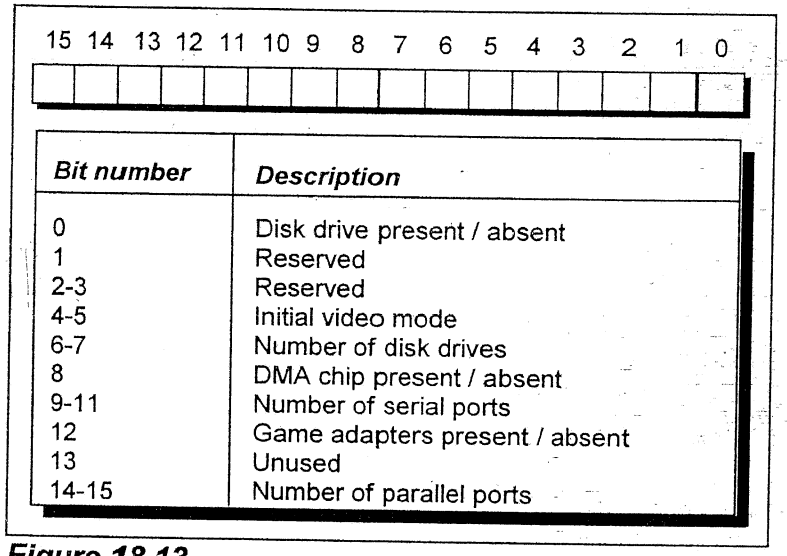
```
/* Display current attribute of a file */

# include <stdio.h>
# include <conio.h>
# include <dos.h>

main( )
{
    char  filename[67] ;
    int  attrib ;
    union REGS  i, o ;
    puts ( "Enter file name " ) ;
    gets ( filename ) ;
    i.h.ah = 67 ;  /* service number - 0x43 */
    i.x.dx = filename ;
    i.h.al = 0 ;
    intdos ( &i, &o ) ;
    if ( o.x.cflag != 0 )  /* carry flag set indicating an Error */
```



```
    {
        puts ( "File not found" ) ;
        goto quit ;
    }
    else
    {
        attrib = o.x.cx ;  /* cx = current file attributes */
        if ( ( attrib & 1 ) == 1 )  /* bit 0 set = read only */
        {
            puts ( "Read only file" ) ;
            goto quit ;
        }
        else
        {
            puts( "Read / Write file" ) ;
            goto quit ;
        }
        if ( ( attrib & 2 ) == 2 )  /* bit 1 set = hidden */
        {
            puts( "Hidden file" ) ;
            goto quit ;
        }
        else
        {
            puts ( "Not a hidden file" ) ;
            goto quit ;
        }
        if ( ( attrib & 4 ) == 4 )  /* bit 2 set = system file */
        {
            puts ( "System file" ) ;
            goto quit ;
        }
        else
        {
            puts ( "Not a system file" ) ;
            goto quit ;
        }
}
```

```
    quit :
        printf ( "\n\n\n\n\nPress any key to exit..." ) ;
        getch( ) ;

}
```

(c) The list of equipment attached to your computer is stored in BIOS Data Area as a two-byte sequence. Using ROM-BIOS interrupt number 17 (refer Appendix H for the details of the service) write a program to display this equipment list on the screen. The bitwise distribution of the two-byte sequence is given below:



Bit number	Description
0	Disk drive present / absent
1	Reserved
2-3	Reserved
4-5	Initial video mode
6-7	Number of disk drives
8	DMA chip present / absent
9-11	Number of serial ports
12	Game adapters present / absent
13	Unused
14-15	Number of parallel ports

Figure 18.13

Program:

```
/* Program to display equipment list on the screen */
```



```
# include <stdio.h >
# include <conio.h>
# include <dos.h>

main( )
{
    union REGS  i, o ;
    unsigned int equiplist, b ;

    int86 ( 17, &i, &o ) ;  /* service number - 0x11 */

    clrscr( ) ;

    equiplist = o.x.ax ;
    printf ( "\nEquipment List :\n" ) ;

    if ( ( equiplist & 1 ) == 1 )  /* bit 0 for disk drive */
    {
        b = ( ( equiplist << 8 ) >> 14 ) ;  /* bit 6 & 7 for number of disk
                                               drives */
        printf ( "\n%d Disk drive present", b + 1 ) ;
    }
    else
        printf ( "\nDisk drive absent" ) ;

    b = ( ( equiplist << 10 ) >> 14 ) ;  /* bit 4 & 5 for initial video mode */
    if ( b == 1 )
        printf ( "\nInitial video mode - 40 x 25 column CGA text" ) ;
    if ( b == 2 )
        printf ( "\nInitial video mode - 80 x 25 column CGA text" ) ;
    if ( b == 3 )
        printf ( "\nInitial video mode - 40 x 25 column MA text" ) ;
    if ( ( equiplist & 256 ) == 256 )  /* bit no. 8 for DMA */
        printf ( "\nDMA chip absent" ) ;
    else
        printf ( "\nDMA chip present" ) ;

    b = ( ( equiplist << 4 ) >> 13 ) ;  /* bit no. 9 to 11 for serial ports */
```

```
    printf ( "\nNo. of serial ports installed = %d", b ) ;

    if ( ( equiplist & 4096 ) == 4096 )  /* bit no. 12 for game adopter */
        printf ( "\nGame adaptor present" ) ;
    else
        printf ( "\nGame adaptor absent" ) ;

    b = equiplist >> 14 ;  /* bit no.14 & 15 for parallel ports */
    printf ( "\nNo. of parallel ports installed = %d", b ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

(d) An animal could be either a canine (dog, wolf, fox etc.), a feline (cat, hynx, jaguar etc.), a catacean (whale, narwhal etc.) or a marsupial (koala, wombat etc.). The information whether a particular animal is canine, feline, catacean, or marsupial is stored in bit number 0, 1, 2 and 3 respectively of a integer variable called **type**. Bit number 4 of the variable **type** stores the information about whether the animal is Carnivore or Herbivore.

For the following animal, complete the program to determine whether the animal is a herbivore or a carnivore. Also determine whether the animal is a canine, feline, catacean or a marsupial.

```
struct animal
{
    char  name[30] ;
    int  type ;
}
struct animal a = { "OCELOT", 18 } ;
```


Program:

```
/* Determine the type of animal */

# include <stdio.h>
# include <conio.h>

main( )
{
    struct animal
    {
        char  name[30] ;
        int  type ;
    } ;
    static struct animal a = { "OCELOT", 18 } ;
    int  ani ;

    clrscr( ) ;
    printf ( "\nAnimal is " ) ;

    ani = a.type ;
    if ( ( ani & 1 ) == 1 )  /* bit 0 */
        printf ( "Canine" ) ;
    if ( ( ani & 2 ) == 2 )  /* bit 1 */
        printf ( "Feline" ) ;
    if ( ( ani & 4 ) == 4 )  /* bit 2 */
        printf ( "Catacean" ) ;
    if ( ( ani & 8 ) == 8 )  /* bit 3 */
        printf ( "Marsupial" ) ;

    printf ( "\nAnimal is also a " ) ;
    if ( ( ani & 16 ) == 16 )  /* bit 4 */
        printf ( "Carnivore" ) ;
    else
        printf ( "Herbivore" ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
```

```
    getch( ) ;
}
```

(e) The time field in the directory entry is 2 bytes long. Distribution of different bits which account for hours, minutes and seconds is given below. Write a function which would receive the two-byte time entry and return to the calling function, the hours, minutes and seconds.

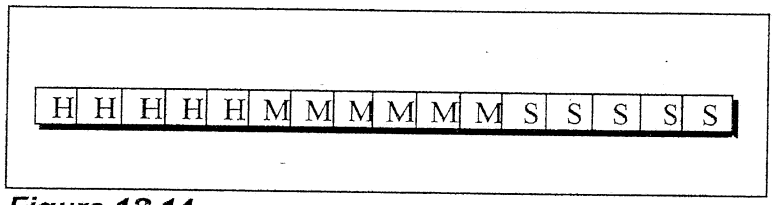


Figure 18.14

Program:

```
/* Program to display hour, minute, and seconds */

#include <stdio.h>
#include <conio.h>

unsigned int  hours, minutes, seconds ;  /* Global variables */

main( )
{
    int  time ;
    void times ( unsigned int time ) ;

    clrscr( ) ;

    puts ( "Enter any  number ( less than 24446 ):" ) ;
    scanf ( "%u", &time ) ;
```



```
    times ( time ) ;
    printf ( "\nFor Time = %u", time ) ;
    printf ( "\nHours = %u", hours ) ;
    printf ( "\nMinutes = %u", minutes ) ;
    printf ( "\nSeconds = %u", seconds ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

void times ( unsigned int  time )
{
    hours = ( time >> 11 ) ;
    minutes = ( ( time << 5 ) >> 10 ) ;
    seconds = ( ( time << 11 ) >> 11 ) * 2 ;
}
```

(f) Using the bit distribution of bytes 0x417 and 0x418 in BIOS Data Area, write a program to display the current status of various shift and toggle keys. Refer Chapter 16, Figure 16.1 for bit distribution of 0x417 and 0x418.

Program:

```
/* Display current status of shift and toggle keys */

#include <stdio.h>
#include <conio.h>
#include <dos.h>

main( )
{
    unsigned char  far *stat = 0x417 ;
    char  status ;

    status = *stat ;
```

```
clrscr( ) ;

if ( ( status & 1 ) == 1 )  /* bit no. 0 */
     printf ( "Right shift pressed" ) ;
else
     printf ( "Right shift not pressed" ) ;

if ( ( status & 2 ) == 2 )  /* bit no 1 */
     printf ( "\nLeft shift pressed" ) ;
else
     printf ( "\nLeft shift not pressed" ) ;

if ( ( status & 4 ) == 4 )  /* bit no. 2 */
     printf ( "\nCtrl pressed" ) ;
else
     printf ( "\nCtrl not pressed" ) ;

if ( ( status & 8 ) == 8 )  /* bit no. 3 */
     printf ( "\nAlt pressed" ) ;
else
     printf ( "\nAlt not pressed" ) ;

if ( ( status & 16 ) == 16 )  /* bit no 4 */
     printf ( "\nScroll lock ON" ) ;
else
     printf ( "\nScroll lock OFF" ) ;

if ( ( status & 32 ) == 32 )  /* bit no. 5 */
     printf ( "\nNum lock ON" ) ;
else
     printf ( "\nNum lock OFF" ) ;

if ( ( status & 64 ) == 64 )  /* bit no 6 */
     printf ( "\nCaps lock ON" ) ;
else
     printf ( "\nCaps lock OFF" ) ;
```



```
     if ( ( status & 128 ) == 128 )  /* bit no 7 */
          printf ( "\nInsert ON" ) ;
     else
          printf ( "\nInsert OFF" ) ;

     printf ( "\n\n\n\n\nPress any key to exit..." ) ;
     getch( ) ;
}
```

(g) What will be the output of the following program:

```
main( )
{
     int i = 32, j = 65, k, l, m, n, o, p ;
     k = i | 35 ; l = ~k ; m = i & j ;
     n = j ^ 32 ; o = j << 2 ; p = i >> 5 ;
     printf ( "\nk = %d l = %d m = %d", k, l, m ) ;
     printf ( "\nn = %d o = %d p = %d", n, o, p ) ;
}
```

Program:

```
/* Output of bitwise operations */

main( )
{
     int i = 32, j = 65, k ;

     clrscr( ) ;

     k = i | 35 ;
     printf ( "k = %d\n", k ) ;  /* k = 35 */

     k = ~k ;
     printf ( "k = %d\n", k ) ;  /* k = -36 */

     k = i & j ;
```

```
    printf ( "k = %d\n", k ) ;  /* k = 0 */

    k = j ^ 32 ;
    printf ( "k = %d\n", k ) ;  /* k = 97 */

    k = j << 2 ;
    printf ( "k = %d\n", k ) ;  /* k = 260 */

    k = i >> 5 ;
    printf ( "k = %d\n", k ) ;  /* k = 1 */

    getch( ) ;
}
```


Chapter 19

The Leftovers

[A] What will be the output of the following programs:

(a)
```
main( )
{
    enum  status { pass, fail, atkt } ;
    enum  status stud1, stud2, stud3 ;
    stud1 = pass ;
    stud2 = fail ;
    stud3 = atkt ;
    printf ( "\n%d %d %d", stud1, stud2, stud3 ) ;
}
```

Output:

0 1 2

(b)
```
main( )
{
    printf ( "%f", ( float ) ( ( int ) 3.5 / 2 ) ) ;
}
```

Output:

1.000000

(c)
```
main( )
{
    float  i, j ;
    i = ( float ) 3 / 2 ;
```

```
    j = i * 3 ;
    printf ( "\n%d", ( int ) j ) ;
}
```

Output:

4

[B] Point out the error, if any, in the following programs:

(a)
```
main( )
{
    typedef struct patient
    {
        char name[20] ;
        int age ;
        int systolic_bp ;
        int diastolic_bp ;
    } ptt ;
    ptt p1 = { "anil", 23, 110, 220 } ;
    printf ( "\n%s %d", p1.name, p1.age ) ;
    printf ( "\n%d %d", p1.systolic_bp, p1.diastolic_bp ) ;
}
```

No Error.

(b)
```
main( )
{
    void show( ) ;
    void ( *s )( ) ;
    s = show ;
    ( *s )( ) ;
}
void show( )
{
    printf ( "\ndon't show off. It won't pay in the long run" ) ;
}
```


No Error.

(c)
```
main( )
{
    int show( ) ;
    int ( *s )( ) ;
    s = show( ) ;
    ( *s )( ) ;
}
float show( )
{
    printf ( "\nControl did reach here" ) ;
    return ( 3.33 ) ;
}
```

Error. Type mismatch in declaration of show, it should be **float show( )** and **float(*s) ( )** ; while assigning the address of the function use s = how.

(d)
```
main( )
{
    void show( int, float ) ;
    void ( *s )( int, float ) ;
    s = show ;
    ( *s )( 10, 3.14 ) ;
}
show ( int i, float f )
{
    printf ( "\n %d %f", i, f ) ;
}
```

Error. Type Mismatch.

[C] Attempt the following:

(a) The list of equipment attached to your computer is stored as a two-byte sequence in BIOS Data Area. Using ROM-BIOS interrupt number 17 write a program to display this equipment list on the screen. Refer Chapter 18, Figure 18.13 for the bitwise distribution of the two-byte sequence.

Hint: Use bit-fields

Program:

```
/* Program to display equipment equiplist on the screen */

# include <stdio.h>
# include <conio.h>
# include <dos.h>

main( )
{
    union REGS  i, o ;
    int  equiplist, b ;
    struct  equip
    {
        unsigned  diskyes : 1 ;
        unsigned  reserved1 : 3 ;
        unsigned  mode : 2 ;
        unsigned  noofdrives : 2 ;
        unsigned  dma : 1 ;
        unsigned  noofsports : 3 ;
        unsigned  game : 1 ;
        unsigned  reserved2 : 1 ;
        unsigned  noofpports : 2 ;
    } ;
    union  reg
    {
```



```
        struct equip e ;
        unsigned int  a ;
    } ;
    union reg eq ;
    clrscr( ) ;

    int86 ( 17, &i, &o ) ;
    eq.a = o.x.ax ;

    printf ( "\nEquipment List :\n" ) ;
    if ( eq.e.diskyes != 0 )
        printf ( "\n%d Disk drive present", eq.e.noofdrives + 1) ;
    else
        printf ( "\nDisk drive absent" ) ;

    if ( eq.e.mode == 1 )
        printf ( "\nInitial video mode - 40 x 25 column CGA text" ) ;
    if ( eq.e.mode == 2 )
        printf ( "\nInitial video mode - 80 x 25 column CGA text" ) ;
    if ( eq.e.mode == 3 )
        printf ( "\nInitial video mode - 40 x 25 column MA text" ) ;

    if ( eq.e.dma == 1 )
        printf ( "\nDMA chip absent" ) ;
    else
        printf ( "\nDMA chip present" ) ;

    printf ( "\nNo. of serial ports installed = %d", eq.e.noofsports ) ;

    if ( eq.e.game == 1 )
        printf ( "\nGame adaptor present" ) ;
    else
        printf ( "\nGame adaptor absent" ) ;

    printf ( "\nNo. of parallel ports installed = %d", eq.e.noofpports ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
```


www.indianfun.org

```
}
```

(b) Write programs which can simulate the following possible virus activities, using the concept of pointer to a function:

(1) Dancing dolls: Every character present on the screen should keep changing from small case to capital irrespective of whether you are working in dbase or wordstar or any other software.

Program:

```
/* Dancing dolls */

/* getvect( ), setvect( ) & keep( ) are Turbo C std. library
     functions */

/* Note: Do not ever try to run any of the TSR program, because it
will not run. So simply execute and create exe file by pressing F9. Then
quite from TC and then run that .exe file */

# include <stdio.h>
# include <stdlib.h>
# include <conio.h>
# include <dos.h>

void interrupt ( *old )( ) ;
void interrupt new( ) ;
char far  *v = ( char far *) 0xB8000000 ;
int  i ;

main( )
{
   old = getvect ( 9 ) ;  /* This function stores the address of a
                             interrupt ( ROM-BIOS routine ) 9 is
                             no. of interrupt issued whenkey is
                             hit */
```



```
   setvect ( 9, new ) ;  /* Here we are setting the interrupt no.
                            9 for our function new. So here
                            onwards whenever key is pressed
                            instead of ROM-BIOS routine our
                            new function gets called */

   keep ( 0, 1000 ) ;  /* This function keeps our program in
                          memory of 1000 bytes, even if the
                          function main gets completed */
}

void interrupt new( )
{
   for ( i = 0 ; i < 4000 ; i + = 2 )
   {
      if ( *( v + i ) >= 65 && *( v + i ) <= 90 )
         *( v + i ) = *( v + i ) + 32 ;
      else if ( *( v + i ) >= 97 && *( v + i ) <= 122 )
         *( v + i ) = *( v + i ) - 32 ;
   }

   (*old)( ) ;  /* After finishing the job which we want to do.
                   Some action of the user calls the ROM-BIOS
                   routine for its normal functioning */
}
```

(2) Rain Rain here again: The characters present on the screen should keep falling down to the bottom of the screen. The character which should start falling should get chosen randomly.

Program:

```
/* Rain Rain Here Again */

/* getvect( ), setvect( ) & keep( ) are Turbo C std. library
     functions */
```

```
# include <stdio.h>
# include <dos.h>
# include <stdlib.h>

void interrupt (*old)( ) ;
void interrupt new( ) ;
char far *v = ( char far *) 0xB8000000 ;
int i, r, c ;
char ch ;

main( )
{
        old = getvect ( 9 ) ;
        setvect ( 9, new ) ;
        keep ( 0, 1000 ) ;
}

void interrupt new( )
{
        r = random (23) ;
        c = random (79) ;
        ch = * ( v + r * 160 + c * 2 ) ;
        for ( i = r ; i <= 23 ; i++ )
        {
            *( v + i * 160 + c * 2 ) = ' ' ;
            *( v + ( i + 1 ) * 160 + c * 2 ) = ch ;
        }
        (*old)( ) ;
}
```

(3) The background color of the screen should keep changing after every 10 seconds.

Program:

```
/* Change background color after every 10 seconds */
```



```
/* getvect( ), setvect( ) & keep( ) are Turbo C std. library
   functions */

#include <dos.h>

void interrupt (*old)( ) ;
void interrupt new( ) ;
char far  *v = ( char far *) 0xB8000000 ;
int  i, ticks, colour = 16 ;

main( )
{
        old = getvect ( 8 ) ;
        setvect ( 8, new ) ;
        keep ( 0, 1000 ) ;
}

void interrupt new( )
{
        ticks++ ;
        if ( ticks > 182 )
        {
            for ( i = 1 ; i < 4000 ; i += 2 )
                * ( v + i ) = colour ;

            ticks = 0 ;
            colour = colour + 16 ;
            if ( colour > 112 )
                colour = 16 ;
        }
        (*old)( ) ;
}
```

(4) Every character sent to the printer is incremented by 1 before it goes to the printer. Thus, if the user tries to print 'Hello' on the printer, your program should see to it that what is sent toS printer is "Ifmmp".

Program:

```
/* Manipulate characters sent to printer */

/* getvect( ), setvect( ) & keep( ) are Turbo C std. library
   functions */

#include <dos.h>

void interrupt (*old)( ) ;
void interrupt new( ) ;

main( )
{
        old = getvect ( 23 ) ;
        setvect ( 23, new ) ;
        keep ( 0, 1000 ) ;
}

void interrupt new( )
{
        if ( _AH == 0 )  /* This _AH is duplicate of CPU Register. If
                            its value is zero then the character is not
                            yet printed on the printer so we are
                            changing that character to its next
                            character and then calling the ROM-
                            BIOS routine */
        _AL = _AL + 1 ;

        (*old)( ) ;
}
```


Chapter 20
Graphics Programming

[A] State True or False:

(a) There are no keywords in C to carry out any drawing activity.

Answer: True

(b) The functions provided in graphics library would vary from one compiler to another.

Answer: True

(c) If we were to carry out drawing under Unix we would have to adapt a different strategy than the one mentioned in this chapter.

Answer: False

(d) Any shape drawn on the screen can be animated using the **getimage( )** & **putimage( )** functions.

Answer: True

(e) Unless we call **initgraph( )** we cannot draw anything on the screen.

Answer: True

(f) Purpose of **initgraph( )** is to load suitable graphics driver.

Answer: True

(g) It is mandatory to call **closegraph( )** at the end of any graphics program.

Answer: True

(h) Using **printf( )** we can write text in a suitable font in graphics mode.

Answer: False

(i) **restorecrtmode( )** restores the pre-**initgraph( )** display mode.

Answer: False

(j) Turbo C's graphics library does not incorporate 3D functionality.

Answer: False

(k) For the character written on the screen in graphics mode there exists an ASCII value and a color Byte in VDU memory.

Answer: False

[B] Answer the following:

(a) Write a program to draw a human shape on the screen and animate it in any of the four directions using the arrow keys.

Program:

```
/* To animate a human shape */

#include <goto.c>
#include <graphics.h>

main( )
{
    int gd = DETECT, gm = DETECT, x, y, ch, i=0, d = 0, area ;
    char *p ;

    initgraph ( &gd, &gm, "c:\\tc\\bgi" ) ; /* initializing graphics */

   /* center coordinates of screen */
```



```
    x = getmaxx( ) / 2;
    y = getmaxy( ) / 2;

    circle ( getmaxx( ) / 2 – 20, getmaxy( ) / 2 – 20, 15 ) ; /* To draw a
                                                                 circle */
    circle ( getmaxx( ) / 2 – 20, getmaxy( ) / 2 + 20, 25 ) ;

    /* To draw a line */
    line ( x-10, getmaxy( ) / 2 + 45, x-10, getmaxy( ) / 2 + 85 ) ;
    line ( x – 28, y + 45, x - 28, y + 85 ) ;

    area = imagesize ( x - 45, y – 35, x + 5, y + 85 ) ;
    p = malloc ( area ) ;
    getimage ( x - 45, y - 35, x + 5, y + 85, p ) ; /* outer rectangle of
                                                       image*/

    outtext ( "Press arrow key to change direction" ) ;

    while ( 1 )
    {
        ch = getkey( );
        switch ( ch )
        {
            case 72: /* Up arrow */

                /* erasing previous and drawing new */

                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                i++ ;
                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                break ;

            case 80: /* Down arrow */

                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                i-- ;
                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                break ;
```


www.indianfun.org

```
            case 77: /* Left arrow */

                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                d-- ;
                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                break ;

            case 75: /* Right arrow */

                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                d++ ;
                putimage ( x – 45 - d, y – 35 - i, p, XOR_PUT ) ;
                break ;

            case 28: /* Enter key */
                exit( );
        }
    }
    getch( ) ;
    closegraph( ) ;
    restorecrtmode( ) ;
}
```

(b) Draw a cube on the screen with three of its faces visible. Ensure that each face is drawn in a different color.

Program:

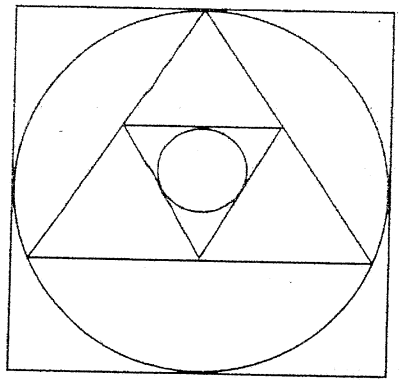
```
/* Cube with three faces visible */

#include <graphics.h>

main( )
{
    int gd = DETECT, gm ;
    int a[ ] = { 250, 150, 200, 200, 300,  200, 350,150, 250, 150 } ;
```



```
    initgraph ( &gd, &gm, "c:\\tc\\bgi" ) ;

    rectangle ( 200, 200, 300, 300 ) ; /*to draw rectangle */
    setfillstyle ( 1, 3 ) ;
    floodfill ( 210, 210, 15 ) ; /* fills the inside of rectangle */

    drawpoly ( 5, a ) ; /* draws a polygon with 5 sides */
    setfillstyle ( 1, 4 ) ;
    floodfill ( 260, 160, 15 ) ;  /* fills the inside of polygon */

    line ( 350, 150, 350, 250 ) ;
    line ( 350, 250, 300, 300 ) ;

    setfillstyle ( 1, 5 ) ;
    floodfill ( 310, 200, 15 ) ; /* fills the rightmost polygon */

    getch( ) ;
    closegraph( ) ;
    restorecrtmode( ) ;
}
```

(c) Write a program to draw the following figure. Each closed region in the figure should be filled with a different color.

Program:



```
/* To draw the given fig. and fill it with different colors */

#include <graphics.h>

main( )
{
    int gd = DETECT, gm, i, j ;
    int a[ ] = { 110, 245, 200, 100, 290, 245, 110, 245 } ;
    int b[ ] = { 150, 180, 250, 180, 200, 245, 150, 180 } ;

    initgraph( &gd, &gm, "c:\\tc\\bgi" ) ;

  /* to draw the outermost rectangle and fill it */
   setcolor ( BLUE ) ;
   rectangle ( 100, 100, 300, 300 ) ;
   setfillstyle ( SOLID_FILL, BLUE ) ;
   floodfill ( 120, 120, BLUE ) ;

  /* to draw the next large circle and fill it */
   setcolor ( RED ) ;
   circle ( 200, 200, 100 ) ;
   setfillstyle ( SOLID_FILL, RED ) ;
   floodfill ( 200, 200, RED ) ;
```



```
  /* to draw the larger triangle and fill it */
   setcolor ( YELLOW ) ;
   drawpoly ( 4, a ) ;
   setfillstyle ( SOLID_FILL, YELLOW ) ;
   floodfill ( 200, 200, YELLOW ) ;

  /* to draw the inner triangle and fill it */
   setcolor ( GREEN ) ;
   drawpoly ( 4, b ) ;
   setfillstyle ( SOLID_FILL, GREEN ) ;
   floodfill ( 200, 200, GREEN ) ;

  /* to draw the innermost small circle and fill it */
   setcolor ( MAGENTA ) ;
   circle ( 200, 204, 24 ) ;
   setfillstyle ( SOLID_FILL, MAGENTA ) ;
   floodfill ( 200, 200, MAGENTA ) ;

   getch( ) ;
   closegraph( ) ;
   restorecrtmode( ) ;
}
```

(d) Write a program to draw a line interactively. Interactive line drawing involves the following:

- Creating a plus shaped cursor.
- Managing the movement of the cursor using arrow keys, Pgup, PgDn, Home and End.
- On positioning the cursor at a suitable place, if enter is hit the point where the cursor was positioned should be treated as the starting point of the line.

- Now the cursor should disappear and as arrow keys, PgUp, PgDn, Home or End are hit the line drawing should begin.
- For every key hit the earlier line should get erased and a new one should get drawn.
- Line should be made permanent when Enter key is hit.

Program:

```
/* Drawing interactive line */

#include <graphics.h>
#include <goto.c>

main( )
{
    int gd = DETECT, gm, x, y, ch, flag = 0, l, h, area ;
    char *p ;

    initgraph ( &gd, &gm, "c:\\tc\\bgi" ) ;

    x = getmaxx( ) / 2 ;
    y = getmaxy( ) / 2 ;

    line ( 0, 2, 4, 2 ) ;
    line ( 2, 0, 2, 4 ) ;

    area = imagesize ( 0, 0, 4, 4 ) ;
    p = malloc ( area ) ;
    getimage ( 0, 0, 4, 4, p) ;

    x = 0 ; y = 0 ;
    l = 0 ; h = 0 ;

    while ( 1 )
    {
```



```
    if ( kbhit( ) )
    {
        if ( flag == 1 )
        {
            putimage ( x - l, y – h, p, XOR_PUT ) ;
            putimage ( x - l, y - h, p, XOR_PUT ) ;
        }
        ch = getkey( ) ;
        switch ( ch )
        {
            case 72 : /* Up arrow */
            case 73 : /* Pageup */

                if ( flag == 1 )
                {
                    putpixel ( x – l , y - h, 12 ) ;
                    h++ ;
                }
                else
                {
                    putimage ( x - l, y - h, p, XOR_PUT ) ;
                    h++ ;
                    putimage ( x - l, y - h, p, XOR_PUT ) ;
                }
                break ;

            case 80 : /* Down arrow */
            case 81 : /* Pagedown */

                if ( flag == 1 )
                {
                    putpixel ( x - l, y - h, 12 ) ;
                    h--;
                }
                else
                {
                    putimage ( x - l, y - h, p, XOR_PUT ) ;
```


www.indianfun.org

```
            h-- ;
            putimage ( x - l, y - h, p, XOR_PUT ) ;
        }

        break;

    case 77 :  /* Left arrow */

        if ( flag == 1 )
        {
            putpixel ( x - l, y - h, 12 ) ;
            l-- ;
        }
        else
        {
            putimage ( x – l, y – h, p, XOR_PUT ) ;
            l--;
            putimage ( x - l, y - h, p, XOR_PUT ) ;
        }
        break;

    case 75 : /* Right arrow */

        if ( flag == 1 )
        {
            putpixel ( x - l, y - h, 12 ) ;
            l++ ;
        }
        else
        {
            putimage ( x - l, y - h, p, XOR_PUT ) ;
            ++l ;
            putimage ( x – l, y - h, p, XOR_PUT ) ;
        }
        break;

    case 28 : /* Enter key */
```



```
                if ( flag == 0 )
                    flag = 1 ;
                else
                    flag = 0 ;
                    break ;

            case 1 : /* Escape key */
                exit( ) ;

            default :
                cleardevice( ) ;
                flag = 0 ;
                putimage ( x - l, y - h, p, XOR_PUT ) ;

            }
        }
    }
    getch( ) ;
    closegraph( ) ;
    restorecrtmode( ) ;
}
```

(e) Write a function to construct a button similar to the one that you often see in Windows. To this function you should pass size of the button, position of the button, color of the button, the text to be displayed on it and the font of the text.

Program:

```
#include <stdio.h>
#include <graphics.h>
#include <string.h>

#define HORIZONTAL HORIZ_DIR
#define VERTICAL VERT_DIR
```


www.indianfun.org

```
main( )
{
    int gd=DETECT, gm, l, r, b, t, color ;
    char  text [ 50 ], align [ 20 ] ;

    initgraph ( &gd, &gm, "C:\\tc\\bgi" ) ;

    outtext ( "enter the size of button in terms of left,top,right and
              bottom" ) ;
    scanf ( "%d%d%d%d", &l, &t, &r, &b ) ;
    cleardevice( ) ;

    outtext ( "enter the color of button (0 to 15)" ) ;
    scanf ( "%d",&color ) ;
    cleardevice( ) ;

    outtext ( "enter the text" ) ;
    fflush ( stdin ) ;
    gets ( text ) ;

    CreateButton ( l, t, r, b, color, text, 0, 1, HORIZONTAL ) ; /* creates
                                                                  button */
    getch( ) ;
    closegraph( ) ;
    restorecrtmode( ) ;
}

int CreateButton ( int left, int top, int right, int bottom, int color, char*
                   text,  int  font, int fontsize, int fontdirn )
{
    int  x, y, l ;
    cleardevice( ) ;

    setcolor ( color ) ;
    rectangle ( left-2, top-2, right+2, bottom+2 ) ; /* outline rectangle of
                                                        button*/
    setfillstyle ( SOLID_FILL, color ) ;
    rectangle ( left, top, right, bottom ) ;
```



```
    floodfill ( ( left + right ) / 2, ( top + bottom ) / 2, color ) ;

    setcolor ( BLACK ) ;
    l = strlen ( text ) ;
    x = ( ( left + right ) / 2 ) - ( l * 4 ) ;
    y = ( top + bottom ) / 2 ;

    settextstyle ( font, 0, fontsize ) ;
    outtextxy ( x - 3l, y, text ) ; /* writes the text on the button at the
                                       specified x,y position*/
}
```

Chapter 21
Mouse Programming

[A] State True or False:

(a) All mouse services have been gather under interrupt number 51.

Answer: True

(b) It is possible to build mouse pointer in graphics mode of size bigger then 16x16 pixel.

Answer: False

(c) It is mandatory that the mouse driver should stand loaded before we can use the mouse.

Answer: True

(d) We can build the arrow shape mouse pointer in text mode.

Answer: False

(e) We can build a color mouse pointer through interrupt number 33h.

Answer: False

(f) To find out the status of the different mouse buttons, we have to use different services under interrupt number 33h.

Answer: True

(g) The service to initialize the mouse also manages to display the mouse pointer.

Answer: False

(h) Once the mouse pointer is displayed, we need to call another service under interrupt number 33h to make the mouse pointer move.

Answer: False

(i) There exist different services to obtain the current mouse position and the status of different mouse buttons.

Answer: False

(j) In any mouse pointer built from 16x16 pixel matrix, the top left pixel is considered to signify the actual mouse position.

Answer: True

[B] Answer the following:

(a) Write a function that would not allow the mouse pointer to enter a particular rectangular area on the screen. Use the following details:

Interrupt number – 33h
AX – 16, service to deactivate mouse when it enters rectangle defined by CX, DX, SI, DI
CX – upper X screen coordinate
DX – upper Y screen coordinate
SI – lower X screen coordinate
DI – lower Y screen coordinate

Program:

```
#include"dos.h"
#include"stdio.h"
#include "conio.h"
#include "graphics.h"

union REGS i,o;

main( )
{
```



```
    int gd, gm, x1, x2, y1, y2 ;
    gd = DETECT ;

    initgraph( &gd, &gm, "c:\\tc\\bgi" ) ;

    rectangle ( 100, 100, 150, 150 ) ;
    initmouse ( ) ;
    while ( ! kbhit ( ) )
    {
        showmouseptr( ); /* shows mouse pointer on the console */
        getmousepos( &x1, &y1, &x2, &y2 ) ;
        while ( x1 > 100  &&  y1 > 100  &&  x1 < 150 && y1 < 150 )
        {
            restrictmouseptr ( 100, 100, 150, 150 ) ;  /* restricts mouse
                                                          pointer */
            getmousepos ( &x1, &y1 ) ;  /* to get the track of mouse */
        }
    }

    getch( ) ;

}

initmouse( )
{
    i.x.ax = 0 ;
    int86 ( 0x33, &i,&o ) ;
    return( o.x. ax ) ;
}
showmouseptr( )
{
    i.x.ax = 1;
    int86( 0x33, &i, &o ) ;
}
restrictmouseptr ( int x1, int y1,int x2, int y2 )
{
    i.x.ax = 16 ;
    i.x.cx = x1 ;
    i.x.dx = y1 ;
    i.x.si = x2 ;
```

```
    i.x.di = y2 ;
    int86( 0x33, &i, &o ) ;
}
getmousepos ( int *x1, int *y1 )
{
    i.x.ax = 3 ;
    int86 ( 0x33, &i, &o ) ;
    *x1 = o.x.cx ;
    *y1 = o.x.dx ;
}
```

(b) Write a function that allows a user to set the sensitivity of the mouse. The sensitivity is measured in a unit called 'mickeys' where 1 mickey represents approximately 1 / 200 of an inch of mouse travel on the table. Effectively it means how much movement of the mouse on the table would cause the mouse cursor to move by one unit on the screen, where one unit corresponds to 8 pixels. Use the following details:

Interrupt number – 33h
AX = 1Ah
BX = horizontal mickeys (1 to 32767, default = 8)
CX = vertical mickeys (1 to 32767, default = 16)

Program:

```
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#include <graphics.h>

union REGS i, o ;
main( )
{

    int gd, gm, x1, x2, y1, y2 ;
    gd = DETECT ;
    clrscr( ) ;
    initgraph ( &gd, &gm, "C:\\tc\\bgi" ) ;
```



```
    initmouse( ) ;
    showmouseptr( ) ;
    printf ( " Enter the value for horizontal mickeys and vertical
            mickeys\n " ) ;
    scanf ( "%d%d", &x1, &y1 ) ;
    setmickey ( x1, y1 ) ;

    getch( ) ;
    closegraph( ) ;
    restorecrtmode( ) ;

}
initmouse( )
{
    i.x.ax = 0 ;
    int86 ( 0x33, &i, &o ) ;
    return ( o.x.ax ) ;
}
showmouseptr( )
{
    i.x.ax = 1;
    int86 ( 0x33, &i, &o ) ;
}
setmickey( int x1, int y1 )
{
    i.x.ax = 0x1a ;
    i.h.bh = x1;
    i.h.ch = y1;
    int86 ( 0x33, &i, &o ) ;
}
```

Chapter 22

C and Assembly

[A] State True or False:

(a) The method of mixing Assembly code in a C language program is known as inline Assembly.

Answer: True

(b) A program written in assembly is always smaller than its C equivalent.

Answer: True

(c) Some instructions in assembly have no counterparts in the C language.

Answer: True

(d) Moving a value directly to a segment register is permitted.

Answer: False

(e) The zero-bit in the flags register is set to 1 if the result is non-zero.

Answer: False

(f) A 16-bit **mul** instruction places the result in AX:DX pair.

Answer: False

(g) Application programs are rarely written completely in Assembly language.

Answer: True

(h) Any offset register can be combined with any segment register to reference a memory location.

Answer: True

[B] Pick up the correct alternative for each of the following:

(a) Assembly code is mixed with C code to

(1) Increase the speed for critical routines
(2) Work closely with hardware
(3) Both 1 and 2
(4) Only 1

Answer

(2) Work closely with hardware

(b) The instruction **sub ax, bx** places the result into

(1) AX register
(2) BX register
(3) Depends upon flag register setting
(4) Memory

Answer

(1) AX register

(c) The instruction **mov [bx], ax** moves the contents of

(1) AX into BX
(2) BX into AX
(3) AX into DS:BX
(4) None of the above

Answer

(3) AX into DS:BX

(d) The video memory in a graphics card is divided into banks of

(1) 64 KB
(2) 64 bytes
(3) 16 KB
(4) None of the above

Answer

(1) 64 KB

(e) If the DS register contains 40h and SI register contains 17h then the DS:SI pair refers to memory location

(1) 417h
(2) 4017h
(3) 40017h
(4) None of the above

Answer

(1)417

Chapter 23
Additional Problems

Constants, Variables and Expressions

[1] Write C expressions corresponding to the following assuming all quantities to be of type float.

(a) $x^5 + 10x^4 + 8x^3 + 4x + 2$

Answer: pow (x , 5) + 10 * pow (x, 4) + 8* pow (x ,3) + 4 * x + 2

(b) ((ax + b) / c) + ((dy + e) / 2f) – (a / bd)

Answer: (((a * x) + b) / c) + (((d * y + e) / 2 * f) – a / b * d)

(c) $A = B_0 \left(\frac{P_2}{\pi} \right)^{\frac{(g+1)}{k}}$

Answer: A = B_0 * pow ((P_2 / 3.14) , (g + 1) / k)

(d) $B = 2\pi \log\left(\sqrt{l/g}\right)\left(\frac{1}{4}\sin^3 A/2 + e^{1.32}\right)$

Answer: B = 2 * 3.14 * log (pow (l / g, 0.5)) *
(¼ * pow (sin (A / 2), 3) + exp (1.32))

(e) $\lambda = \sqrt{4D^2(\beta-\omega)^2} + \tan^{-1}\frac{(\delta+\mu)}{2C}$

Answer: λ = sqrt (4 * pow (D ,2) – pow ((β -ω), 2)) + atan ((δ + μ) / 2 * C)

(f) $V = 2\nabla\mu r\left[eA + \frac{R}{l}\log A\sec A + \frac{1}{\varphi}\left(\frac{r}{\gamma^3}\right)\sin^4 A\cos\theta\right]$

Answer: V = 2 * ∇ * μ * r (e * A + R / l * log (A) * sec (A) + 1 / φ (r / pow (γ , 3)) * pow (sin (A) , 4) * cos (θ))

[2] Evaluate the following expressions:

float a = 2.5, b = 2.5 ;

(a) a + 2.5 / b + 4.5

Answer: 8.000000

(b) (a + 2.5) / b + 4.5

Answer: 6.500000

(c) (a + 2.5) / (b + 4.5)

Answer: 0.714286

(d) a / 2.5 / b

Answer: 0.400000

(e) b++ / a + b--

Answer: 4.500000

[3] Evaluate each of the following expressions (The expressions are to be evaluated independent of one another):

int i = 3, j = 4, k = 2 ;

(a) i * = k = ++j + i

Answer: 24

(b) i = j /= k + 4

Answer: 0

[4] Classify the following constants as decimal, octal or hexadecimal:

(a) 01234

Answer: octal

(b) –02456

Answer: octal

(c) 0x1111

Answer: hexadecimal

(d) –128734689

Answer: decimal

(e) –0xaAbB

Answer: hexadecimal

(f) 122

Answer: decimal

[5] The ideal compressor outlet temperature and efficiency for a gas turbine are given by

$$T = T_0\left(\frac{P_2}{P_1}\right)^{\frac{(g-1)}{g}}$$

and

$$e = 1 - \left(\frac{P_1}{P_2}\right)^{\frac{(g-1)}{g}}$$

where $\mathbf{T_0}$ is the inlet temperature, $\mathbf{P_1}$ the inlet pressure, $\mathbf{P_2}$ the outlet pressure, and g the ratio of specific heats. Write a program to compute and Output T and e for $\mathbf{T_0}$ = 450, $\mathbf{P_1}$ = 10, $\mathbf{P_2}$ = 55 and **g** = 1.5.

Program:

```
#include <math.h>
main( )
{
    float  T0 = 450, P1 = 10, P2 = 55, g = 1.5, e, T ;

    clrscr( ) ;

    T = T0 * pow ( P2 / P1, ( g - 1 ) / g ) ;
    e = 1 - pow ( P1 / P2, ( g - 1 ) / g ) ;

    printf ( "T = %f\te = %f", T, e ) ;
```



```
    getch( ) ;
}
```

Control Instructions

[1] Given a quadratic equation $\mathbf{ax^2 + bx + c = 0}$. Write a program to find the roots of it.

Program

```
/* Roots of a quadratic equation */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    float  a, b, c, x ;
    float  y, r, r1, r2 ;
    clrscr( ) ;
    printf ( "Enter the values of a, b and c:\n" ) ;
    scanf ( "%f%f%f", &a, &b,&c ) ;

    y = ( b * b - ( 4 * a * c ) ) ;

    if ( y >= 0 )
    {
        r = sqrt ( y ) ;
        r1 = ( - (b * b ) + r ) / ( 2 * a ) ;
        r2 = ( - (b * b ) - r ) / ( 2 * a ) ;
        printf ( "\nRoot1 = %.2f\tRoot2 = %.2f", r1, r2 ) ;
    }
    else
    {
        printf ( "\nComplex roots:" ) ;
```

```
        r = sqrt ( -y ) ;
        r1 = -( b * b) / ( 2 * a ) ;
        r2 = r1 ;
        printf ( "\nRoot1 = %.2f + i %.2f", r1, r ) ;
        printf ( "\nRoot2 = %.2f - i %.2f", r2, r ) ;
    }
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

[2] Given fifty pairs of length and breadth of rectangle, write a program to find all the rectangles whose area its grater than their perimeters. For example, the area of the rectangle with length = 5 and breadth = 4 is greater than its perimeter.

Program

```
/* Compare Area And Perimeter of rectangle */

#include <stdio.h>
#include <conio.h>

main( )
{
    int len, brth, i ;
    float area, peri ;

    for ( i = 0 ; i < 50 ; i++ )
    {
        clrscr( ) ;

        printf ( "\n\nEnter the length and breadth of the Rectangle:\n" ) ;
        scanf ( "%d%d", &len, &brth ) ;
        area = len * brth ;
        peri = 2 * ( len + brth ) ;
        if ( area > peri )
        {
```



```
            printf ( "\nAREA is greater than the PERIMETER of the
                    Rectangle." ) ;
            printf ( "\nArea is %.2f  and  Perimeter is %.2f  ", area,
                    peri ) ;
            printf ( "\nlength = %d  and  Breadth = %d", len, brth ) ;
        }
        else
        {
            if ( area == peri )
            {
                printf ( "\nPerimeter is equal to Area of the
                        Rectangle" ) ;
                printf ( "\nArea is %.2f  and  Perimeter is %.2f  ",
                        area, peri ) ;
            }
            else
            {
                printf ( "\nPerimeter is greater than Area of the
                        Rectangle" ) ;
                printf ( "\nArea is %.2f and Perimeter is %.2f ", area,
                        peri ) ;
            }
        }
        getch( ) ;
    }
}
```

[3] Given three point **(x1, y1)**, **(x2, y2)** and **(x3, y3)**, write a program to check if they are collinear:

Program:

```
/* To check colinear points */

#include <stdio.h>
#include <conio.h>
#include <math.h>
```


www.indianfun.org

```
main( )
{
    int x1, y1, x2, y2, x3, y3;
    int s1, s2, s3 ;

    clrscr( ) ;
    printf ( "\nEnter the values of x1 and y1 coordinates of first point:" ) ;
    scanf ( "%d%d", &x1, &y1 ) ;
    printf ( "\nEnter the values of x2 and y2 coordinates of first point: " ) ;
    scanf ( "%d%d", &x2, &y2 ) ;
    printf ( "\nEnter the values of x3 and y3 coordinates of first point: " ) ;
    scanf ( "%d%d", &x3, &y3 ) ;

    /* Calculate Slope of line between each point */
    s1 = abs ( x2- x1 ) / abs ( y2 - y1 ) ; /* Slope between 1 st & 2 nd
                                               point */
    s2 = abs ( x3- x1 ) / abs ( y3 - y1 ) ; /* Slope between 1 st & 3 rd
                                               point */
    s3 = abs ( x3- x2 ) / abs ( y3 - y2 ) ; /* Slope between 2 nd & 3 rd
                                               point */

    if ( ( s1 == s2 ) && ( s1 == s3 ) )
        printf ( "\n\tPoints are Collinear" ) ;
    else
        printf ( "\n\tPoints are NOT Collinear" ) ;
    getch( ) ;
}
```

[4] Given the coordinates **(x, y)** of a center of a circle and it's radius, write a program which will determine whether a point lies inside the circle, on the circle or outside the circle.

Program:

```
/* Check if point lies within a circle */
/* The centre of the circle has been assumed to be at 0,0 */
#include <stdio.h>
```



```
#include <conio.h>

main( )
{
    int x, y, r ;
    int dis, d ;
    clrscr( ) ;
    printf ( "\nEnter the radius of the circle and \nthe coordinates of the
              point ( x, y ):\n" ) ;
    scanf ( "%d\n%d\n%d", &r, &x, &y ) ;
    dis = x * x + y * y ;
    d = r * r ;
    if ( dis == d )
        printf ( "\nPoint with ( %d, %d ) coordinates is on the circle",
                 x, y ) ;
    else
    {
        if ( dis > d )
            printf ("Point with (%d,%d) coordinates is outside the
                     circle", x, y ) ;
        else
            printf ("Point with (%d,%d) coordinates is inside the circle",
                     x, y ) ;
    }
    getch( ) ;
}
```

[5] A machine is purchased which will produce earning of Rs. 1000 per year while it lasts. The machine costs Rs. 6000 and will have a salvage of Rs. 2000 when it is condemned. If 12 percent per annum can be earned on alternate investments what would be the minimum life of the machine to make it a more attractive investment compared to alternative investment?

Program:

```
/* What would be the minimum life of the machine */

#include <stdio.h>
#include <conio.h>

main( )
{
    int year = 0, i_cost = 6000, profit = 1000 ;
    int s_cost ;
    float t1, t2, interest ;
    int i ;

    clrscr( ) ;

    t2 = i_cost * 1.0 ;

    while ( 1 )
    {
        year++ ;
        t1 = i_cost + profit * year ;
        interest = t2 * 0.12 ;
        t2 = t2 + interest ;
        printf ( "\nt1 = %f\tt2 = %f + %f = %f", t1, t2 - interest, interest,
                 t2 ) ;
        if ( ( t2 - 2000 ) > ( t1 - 2000 ) )
        {
            year-- ;
            break ;
        }
    }
    printf ( "\nMinimum life of the Machine is %d Years.", year ) ;
    getch( ) ;
}
```


[6] Given a point **(x, y)**, write a program to find out if it lies on the x-axis, y-axis or at the origin, viz. (0, 0).

Program:

```
/* Determine where a point lies */

#include <stdio.h>
#include <conio.h>

main( )
{
    int x, y ;
    clrscr( ) ;
    printf ( "\nEnter the x and y coordinates of a point:\n" ) ;
    scanf ( "%d%d", &x, &y ) ;
    if ( x == 0 && y == 0 )
        printf ( "Point lies on origin" ) ;
    else
        if ( x == 0 && y != 0 )
            printf ( "\nPoint lies on Y axis" ) ;
        else
            if ( x != 0 && y == 0 )
                printf ( "\nPoint lies on X axis" ) ;
            else
                printf ( "\nPoint dosen't lie on any axis, nor origin" ) ;
    getch( ) ;
}
```

[7] Extend the above program to find whether it lies in the first, second, third or fourth quadrant in x-y plane.

Program:

```
/* Which quadrant does a point lie in */
```

```
#include <stdio.h>
#include <conio.h>

main( )
{
    int x, y ;

    clrscr( ) ;

    printf ( "\nEnter the x and y coordinates of a point: " ) ;
    scanf ( "%d%d", &x, &y ) ;

    if ( x == 0 && y == 0 )
        printf ( "\nPoint lies on the origin" ) ;
    if ( x == 0 && y != 0 )
        printf ( "\nPoint lies on Y axis" ) ;
    if ( x != 0 && y == 0 )
        printf ( "\nPoint lies on X axis" ) ;

    if ( x > 0 && y > 0 )
        printf ( "\nPoint lies in the FIRST Quadrant" ) ;
    if ( x < 0 && y < 0 )
        printf ( "\nPoint lies in the THIRD Quadrant" ) ;
    if ( x > 0 && y < 0 )
        printf ( "\nPoint lies in the FORTH Quadrant" ) ;
    if ( x < 0 && y > 0 )
        printf ( "\nPoint lies in the SECOND Quadrant" ) ;

    printf ( "\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

[8] A university has the following rules for a student to qualify for a degree with A as the main subject and B as the subsidiary subject:

(a) He should get 55 percent or more in A and 45 percent or more in B.

(b) If he gets than 55 percent in A he should get 55 percent or more in B. However, he should get at least 45 percent in A.

(c) If he gets less than 45 percent in B and 65 percent or more in A he is allowed to reappear in an examination in B to qualify.

(d) In all other cases he is declared to have failed.

Write a program to receive marks in A and B and Output whether the student has passed, failed or is allowed to reappear in B.

Program:

```
/* University passing grade */

#include <stdio.h>
#include <conio.h>

main( )
{
    int a, b ;

    clrscr( ) ;

    printf ( "\nEnter marks of subject A and B: " ) ;
    scanf ( "%d%d", &a, &b ) ;

    if ( ( (a >= 55 ) && ( b >= 45 ) ) || ( ( ( a >= 45 ) && ( a < 55 ) ) &&
       ( b >= 55 ) ) )
        printf ( "\nThe student has passed in the examination." ) ;
    else
        if ( ( b < 45 ) && ( a >= 65 ) )
            printf ( "\nThe student can reappear for subject B." ) ;
        else
            printf ( "\nThe student has failed in the examination." ) ;
```

```
    getch( ) ;
}
```

[9] The policy followed by a company to process customer orders is given by the following rules:

(a) If a customer order is less than or equal to that in stock and has credit is OK, supply has requirement.
(b) If has credit is not OK do not supply. Send him intimation.
(c) If has credit is Ok but the item in stock is less than has order, supply what is in stock. Intimate to him data the balance will be shipped.

Write a C program to implement the company policy.

Program:

```
/* Company policy for customer orders */

#include <stdio.h>
#include <conio.h>

main( )
{
    int s = 100, o ;
    char c = 'o' ; /* credit is ok */

    clrscr( ) ;
    printf ( "\nEnter the number of items to be ordered:" ) ;
    scanf ( "%d", &o ) ;

    if ( o <= s )
    {
        if ( c == 'o' )
            printf ( "\nOrder is supplied" ) ;
        else
            printf ( "\nOrder cann't be supplied" ) ;
```



```
    }
    else
    {
        if ( c == 'o' )
            printf ( "\nLimited items are supplied, rest would be
                shipped" ) ;
        else
            printf ( "\nOrder cann't be supplied" ) ;
    }

    getch( ) ;
}
```

[10] When interest, compounds **q** times per year at an annual rate of **r**% for **n** years, the principle **p** compounds to an amount **a** as per the following formula

$$a = p (1 + r / q)^{nq}$$

Write a program to read 10 sets of **p, r, n** & **q** and calculate the corresponding as.

Program:

```
/* Compound interest calculation */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    float q, r, n, p, a ;
    int i ;
    for ( i = 0 ; i< 1 ; i++ )
    {
        clrscr( ) ;
```


www.indianfun.org

```
        printf ( "Enter the principal amount:" ) ;
        scanf ( "%f", &p ) ;
        printf ( "Enter the rate of interest:" ) ;
        scanf ( "%f", &r ) ;
        printf ( "Enter the number of years: " ) ;
        scanf ( "%f", &n ) ;
        printf ( "Enter the compounding period: " ) ;
        scanf ( "%f", &q ) ;

        a = p + pow ( (1 + ( r / q ) ), ( n / q ) ) ;

        printf ( "\n\nTotal amount = %f", a ) ;
        getch( ) ;
    }
}
```

[11] A projectile fired at an angle θ has a horizontal range **R** given by

$$R = \frac{V^2 2 \sin\theta \cos\theta}{g}$$

Where V = initial velocity and g = 32.2 ft/sec2. Write a program to compute **R** for the following set of values.

V	θ
200	20
200	70
200	45
175	60
179	75
210	20

Program:

```
/* Calculate distance traveled of a projectile */
```



```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define PI 3.1415922

main( )
{
    int i ;
    double V, R, x, g = 32.2 ;

    clrscr( ) ;

    for ( i = 0; i <= 5; i++ )
    {
        printf ( "\nEnter the initial Velocity and the Angle: " ) ;
        scanf ( "%lf%lf", &V, &x ) ;
        x = x * PI / 180 ; /* Convert to radians */
        R = ( 2 * V * V * sin ( x ) * cos ( x ) ) / g ;
        printf ( "\n\tHorizontal Range = %.3lf", R ) ;
        getch( ) ;
    }
}
```

[12] The period **P** of a pendulum is given by the following formula.

$$P = 2\pi \sqrt{Vg} \left(\frac{1}{4} \sin^2 \alpha/2 + 1 \right)$$

where g = 980 cm/sec2 1= pendulum length, α = angle of displacement.
Write a program to read pendulum number, **I** and α for 10 different pendulums and compute the period of each pendulum. Also print the pendulum number whose period is smallest.

Program:

```
/* Compute period of pendulum */
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define PI 3.1415922

main( )
{
    int  c = 0, i ;
    float  t = 32767, a, l, p, g = 980 ;

    clrscr( ) ;
    for ( i = 1 ; i<=3 ; i++ )
    {
        printf ( "\nEnter the length and angle of displacement of %d
                  pendulum:\n", i ) ;
        scanf ( "%f%f", &l, &a ) ;
        a = a * PI / 180 ;  /* Convert to radians */
        p = 2 * PI * sqrt ( l / g ) * ( 1.0 / 4 * pow ( sin( a / 2 ) , 2 ) + 1 ) ;

        printf ( "\nperiod of pendulum is %f\n", p ) ;
        if ( t >= p )
        {
            t = p ;
            c = i ;
        }
        getch( ) ;
    }
    printf ( "\n\nSmallest period is %f of pendulum number %d", t, c ) ;
    getch( ) ;
}
```

[13] A tank having the shape of an inverted right circular cone of radius **R** and height **h** is initially filled with water. At the bottom of the tank is a hole of radius though which water drains under the influence of gravity. The equation for the time **t** it takes to empty is

$$t = \frac{R^2}{5r^2}\frac{\sqrt{2h}}{\sqrt{g}}$$

Where **g** is the gravitational constant 32.17 ft / sec^2. Compute the time to empty tanks of the following dimensions.

Tank height	Tank radius	Hole radius
6.5	4.0	0.2
10.35	5.5	1.0
15.65	6.0	4.0
17.0	4.9	2.5
14.55	7.5	2.0
16.65	5.57	0.5

Also determine which tank would be emptied earliest.

Program:

```
/* Compute time to empty tank */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    int  i, tank = 0;
    float  R, r, h, t, tt = 99999, g = 32.17 ;

    clrscr( ) ;

    for ( i = 0 ; i<= 5 ; i++ )
    {
        printf ( "\nEnter the Radius and height of tank: " ) ;
        scanf ( "%f%f", &h, &R ) ;
```

```
        printf ( "Enter the Radius of Hole: " ) ;
        scanf ( "%f", &r ) ;

        t = ( R * R * sqrt ( 2 * h ) ) / ( 5 * r * r * sqrt ( g ) ) ;

        printf ( "\tTime to empty the tank = %f", t ) ;
        if ( tt > t )
        {
            tt = t ;
            tank = i ;
        }
    }
    printf ( "\n\nMinimum time required from the 6 tanks is = %f", tt ) ;
    printf ( "\nThe Tank number is = %d", tank ) ;
    getch( ) ;
}
```

[14] The length **L** of a belt needed to wrap around two pulleys of diameters **D** and **d** and whose centers are separated by a distance **C** is given by,

$$L = \sqrt{4C^2 - (D-d)^2} + \pi\frac{(D+d)}{2} + (D-d)\sin^{-1}\frac{(D-d)}{2C}$$

Write a program to compute belt length for each of the following sets of data:

D	d	C
22	8	60
20	10	18
21	11	15
19	10	14
18	12	16

Program:

```
/* Compute lengths of belt */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    int i, D, d, C ;
    float L ;

    clrscr( ) ;

    for ( i = 0 ; i<= 4 ; i++ )
    {
        printf ( "\nEnter the values of D, d and C: " ) ;
        scanf ( "%d%d%d", &D, &d, &C ) ;

        L = sqrt ( 4 * C * C – pow ( D - d, 2 ) ) + 3.14 * ( D + d ) / 2 +
            (D - d) * ( asin ( ( D – d ) / ( 2 * C ) * 3.1415299 / 180 ) ) ;
        printf ( "Length of the belt = %f\n", L ) ;
        getch( ) ;
    }
}
```

[15] A piston is connected to a crankshaft of radius **r** by a connecting rod of length **l**. The velocity v of the piston as a function of the crankshaft angle **A** and revolutions per minute **N** of the shaft is give by,

$$V = 2\pi Nr[\sin A + \frac{r}{l}\sin A\cos A + \frac{1}{2}\left(\frac{r}{l^3}\right)\sin^3 A\cos A]$$

Write a program to compute the velocity in meters/sec for each of the following.

N	A	l	r
3000	68	0.75	0.25
4000	95	0.75	0.2
3300	180	0.95	0.1
4500	90	0.6	0.15
800	270	0.5	0.15
990	75	0.7	0.25

Program:

```
/* Compute velocity of a piston */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    float i, r, l, A, N ;
    float v ;

    clrscr( ) ;

    for ( i = 0 ; i <= 5 ; i++)
    {
        printf ( "\nEnter the number of revolutions per minute: " ) ;
        scanf ( "%f", &N ) ;

        printf ( "Enter the crankshaft angle: " ) ;
        scanf ( "%f", &A ) ;

        printf ( "Enter the length of the con-rod in meters: " ) ;
        scanf ( "%f", &l ) ;
```



```
        printf ( "Enter the radius of the crankshaft in meters: " ) ;
        scanf ( "%f", &r ) ;

        A = A * 3.1415299 / 180 ;
        v = 2 * 3.14 * N * r * ( sin ( A ) + r / l * sin ( A ) * cos ( A ) +
            1.0 / 2 * ( r / pow ( l, 3 ) ) * pow ( sin ( A ), 3 ) * cos ( A ) ) ;
        printf ( "\nVelocity of the piston in meters/sec is = %f
                 meters/sec\n", v ) ;

        getch( ) ;
    }
}
```

[16] The natural logarithm can be approximated by the following series.

$$\frac{x-1}{x}+\frac{1}{2}\left(\frac{x-1}{x}\right)^2+\frac{1}{2}\left(\frac{x-1}{x}\right)^3+\frac{1}{2}\left(\frac{x-1}{x}\right)^4+\ldots.$$

If **x** is input through the keyboard, write a program to calculate the sum of first seven terms of this series.

Program:

```
/* Compute natural logarithm */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    int x, i ;
    float result = 0 ;

    clrscr( ) ;
```

```
    printf ( "\nEnter the value of x: " ) ;
    scanf ( "%d", &x ) ;

    for ( i = 1; i <= 7 ; i++ )
    {
        if ( i == 1 )
            result = result + pow ( ( x - 1.0 ) / x, i ) ;
        else
            result = result + ( 1.0 / 2 ) * pow ( ( x - 1.0 ) / x, i ) ;
    }
    printf ( "\nLog(%d) = %f", x, result ) ;

    getch( ) ;
}
```

[17] The following formulae describe problems of portions of circle, with radius **r**, and central angle θ in degrees.

$$Area = \pi r^2$$

$$Length\ of\ arc = \pi r \frac{\theta}{180}$$

$$length\ of\ chord = 2 r \sin(\theta/2)$$

$$area\ of\ sagment = \frac{\pi r^2 \theta}{360} - \frac{r^2 \sin\theta}{2}$$

If values of **r** and θ are entered through the keyboard, write a program to calculate the above properties of the circle. Note that your problem must work till the user keeps on supplying the values of **r** and θ.

Program:

```
/* Compute Properties of Circle */

#include <stdio.h>
#include <conio.h>
```



```
#include <math.h>
#define PI 3.1415922

main( )
{
    int ang , r ;
    float loarc, loc, aos, area ;
    char ch = 'y' ;

    clrscr( ) ;

    while( ch == 'y' || ch == 'Y' )
    {
        printf ( "\nEnter the values of radius and the angle: " ) ;
        scanf ( "%d%d", &r, &ang ) ;

        area = PI * r * r ;
        loarc = PI * r * ( ang / 180.0 ) ;
        loc = 2.0 * r * sin ( ( ang * 3.1415299 / 180 ) / 2.0 ) ;
        aos = ( PI * r * r * ang ) / 360 - ( r * r * sin ( ang * 3.1415299 /
              180 ) ) / 2 ;

        printf ( "\nArea = %f", area ) ;
        printf ( "\nlenght of arc = %f", loarc ) ;
        printf ( "\nlenght of chord = %f", loc ) ;
        printf ( "\nArea of segment = %f", aos ) ;
        printf ( "\n\nDo you want to continue ? (y/n) " ) ;

        fflush ( stdin ) ;
        scanf ( "%c", &ch ) ;
    }
}
```

[18] A projectile fired at an angle θ with an initial velocity Vo will travel a distance **r** according to the following formula:

$$r = \frac{V_0^2 \sin 2\theta}{g}$$

Where **g** is the acceleration = 9.81 m/sec 2

It will be in motion foe a time **t** given by

$$t = \frac{V_0 \sin\theta}{g}$$

and reach a maximum height **h** given by.

$$h = \frac{V_0^2 \sin\theta}{g}$$

For all angles from 20 to 70 degrees in 10-degree increments and initial velocity of 50 m/sec write a program to prepare the following table.

Projectile Characteristics

Angle	Range	Time	Height
20	-	-	-
30	-	-	-
40	-	-	-
50	-	-	-
60	-	-	-
70	-	-	-

Program:

```
/* Projectile motion */

#include <stdio.h>
#include <conio.h>
#include <math.h>
```



```
#define PI 3.1415922
main( )
{
    double  v = 50.0, r, t, h, g = 9.81 ;
    int  a = 20 ;

    clrscr( ) ;

    printf ( "\n\t\t    Projectile Characteristics\n " ) ;
    printf ( "\n\tAngle\t  Range\t  Time\t\tHeight\n\n " ) ;

    for ( ; a <= 70 ; a += 10 )
    {
        r = ( v * v * sin ( 2 * a * ( PI / 180 ) ) ) / g ;
        t = ( v * sin ( a * ( PI / 180 ) ) ) / g ;
        h = ( v * v * sin ( a * ( PI / 180 ) ) ) / g ;
        printf ( "\n\t%d\t%lf\t%lf\t%lf", a, r, t, h ) ;
    }
    getch( ) ;
}
```

[19] The heat flow **Q** from an insulated pipe is given by the following formula:

$$Q = \frac{2\pi\ KL(T_i - T_0)}{\log_e(D_2 / D_1)}$$

where **L** = pipe length, **Ti** = Temperature inside, $\mathbf{T_0}$ = Temperature outside, **D2** = Outer diameter of pipe (with insulation), **D1** = Inner diameter of pipe (without insulation) and **K** = Thermal conductivity factor of the insulation.
Assume a pipe 50 feet long with a diameter of 8 inches and a thermal conductivity factor 0.035. The outer temperature should be 150 degrees and the inner temperature should be 850 degrees.

Write a program to construct a table of heat flow versus thickness of insulation in half-inch increments from 1 to 4 inches.

Program:

```
/* Heat flow from an insulated pipe */

#include <stdio.h>
#include <conio.h>
#include <math.h>
#define PI 3.1415922

main( )
{
    float  q , k=0.035, l = 50 , t=850, t0=150, d2, d1=8, c = 1.0 ;

    clrscr( ) ;

    printf ( "\n\tHeat flow\tThickness\n" ) ;

    for ( ; c <= 4 ; c += 0.5 )
    {
        d2 = d1 + c ;
        q = ( 2 * PI * k * l * ( t - t0 ) ) / ( log ( d2 / d1 ) ) ;
        printf ( "\n\t%.6f\t%.6f\n", q, c ) ;
    }
    getch( ) ;
}
```

[20] For a cylindrical tube of outside diameter $\mathbf{D_0}$ and an inside diameter **Di**, the moment of inertia is,

$$\frac{\pi (D_0^4 - D_i^4)}{64}$$

and the radius of gyration

$$\frac{\sqrt{D_0^2 - D_i^2}}{4}$$

The external tube diameter and initial thickness are input through the keyboard. Prepare a table for moment of inertia and radius of gyration, incrementing the thickness by a step given by the program user. Terminate the program when the tube becomes a solid.

Program:

```
/* Moment of inertia & Radius of gyration of a tube */

#include <stdio.h>
#include <conio.h>
#include <math.h>
#define PI 3.1415922

main( )
{
    float  i, moi, rog, d0, di, t ;
    clrscr( ) ;

    printf ( "\nEnter the Diameter of the tube: " ) ;
    scanf ( "%f", &d0 ) ;

    printf ( "\nEnter the initial thickness of the tube: " ) ;
    scanf ( "%f", &t ) ;

    printf ( "\nEnter the value for incrementation of the thickness: " ) ;
    scanf ( "%f", &i ) ;

    printf ( "\nThickness\tMoment of Inertia\tRadius of Gyration " ) ;
```

```
    while( 1 )
    {
        di = d0 - t * 2 ;
        moi = PI * ( pow ( d0, 4 ) – pow ( di, 4 ) ) / 64 ;
        rog = ( sqrt ( pow ( d0, 2 ) – pow ( di, 2 ) ) ) / 4 ;
        if ( di == 0 )
            break ;

        printf ( "\n%f\t%f\t\t%f", t, moi, rog ) ;
        t = t + i ;
    }
    getch( ) ;
}
```

[21] In a crankshaft, connecting rod and piston arrangement, the displacement **D** of the piston is given by

$$D = r\left[1 - \cos A + \frac{r}{2l}\sin^2 A + \left(\frac{r}{2l}\right)^3 \sin^4 A\right]$$

Prepare a table of displacement versus the crankshaft angle from 0 to 360 degrees in uniform increment of 10 degrees. The program should read the values of **r** and **l**.

Program:

```
/* Table of displacement Vs Angle of a crankshaft */

#include <stdio.h>
#include <conio.h>
#include <math.h>
#define PI 3.1415922

main( )
{
    float d, a, l, r, i ;
```



```
    clrscr( ) ;
    printf ( "Enter the value of radius of shaft: " ) ;
    scanf ( "%f", &r ) ;
    printf ( "\nEnter the lenght of shaft: " ) ;
    scanf ( "%f", &l ) ;
    printf ( "\n\tCrankshaft Angle\tDispiacement\n" ) ;

    for ( i = 0 ; i <= 360 ; i+= 10 )
    {
        d = r * ( 1 – cos ( i * ( PI / 180 ) ) + r / ( 2 * l ) * pow ( sin ( i* ( PI /
            180 ) ), 2 ) + pow ( ( r / ( 2 * l ) ), 3 ) *
            pow ( sin ( i * ( PI / 180 ) ), 4 ) ) ;
        if ( i == 180 )
        {
            getch( ) ;
            clrscr( ) ;
            printf ( "\n\tCrankshaft Angle\tDisplacement\n" ) ;
        }
        printf ( "\n\t%f\t\t%.10f", i, d ) ;
    }
    getch( ) ;
}
```

Functions

[22] Write function to add, subtract, multiple and divide two complex numbers **(x + iy)** and **(a + ib)**.

Program:

```
/* Addition, Subtraction, multiplication & Division of Complex Numbers */

#include <stdio.h>
#include <conio.h>

main( )
{
```

```
    int x, y, a, b ;

    clrscr( ) ;

    printf ( "Enter four real numbers:" ) ;
    scanf ("%d%d%d%d", &x, &y, &a, &b ) ;
    add ( x, y, a, b ) ;
    sub ( x, y, a, b ) ;
    mult ( x, y, a, b ) ;
    div ( x, y, a, b ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* Function to add complex numbers */
add ( int x, int y, int a, int b )
{
    int r, i ; /* r = real part, i = imaginary part */
    r = x + a ;
    i = y + b ;
    printf ( "\nThe sum of the complex numbers is %d + i(%d)", r, i ) ;
    return ;
}

/* Function to subtract complex numbers */
sub ( int x, int y, int a, int b )
{
    int r, i ;
    r = x - a ;
    i = y - b ;
    printf ( "\nThe substraction of the complex numbers is %d + i(%d)",
              r, i ) ;
    return ;
}

/* Function to multiply complex numbers */
mult ( int x, int y, int a, int b )
```



```
{
    int r, i ;
    r = ( x * a ) - ( y * b ) ;
    i = ( x * b ) + ( y * a ) ;
    printf ("\nThe multiplication of the complex numbers is %d + i(%d)",
              r, i ) ;
    return ;
}

/* Function to divide complex numbers */
div (int x, int y, int a, int b )
{
    int r, i, q ;
    r = x * a + y * b ;
    i = - ( x * b ) + y * a ;
    q = a * a + b * b ;
    printf ( "\nThe division of the complex numbers is %.3f + i ( %.3f )",
              r * 1.0 / q, i*1.0 / q ) ;
    return ;
}
```

[23] Write a C function to evaluate the series

$$\sin(x) = x - (x^3 / 3!) + (x^5 / 5!) - (x^7 / 7!) + \cdots$$

to five significant digits.

Program:

```
/* Evaluation of a series */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    float n, x, a, b, sum = 0 ;
```

```
    int i, j ;
    float numerator ( float, int ), denominator ( int ) ;

    clrscr( ) ;

    printf ( "Enter the number x: " ) ;
    scanf ( "%f", &x ) ;
    for ( i = 1, j = 1 ; j <= 19 ; i++, j += 2 ) /* upto 10 terms */
    {
        a = numerator ( x, j ) ;
        b = denominator ( j ) ;
        n = a / b ;
        ( i % 2 == 0 ) ? sum = sum - n : ( sum = sum + n ) ;
    }
    printf ( "%f", sum ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* Calculate Power */
float numerator ( float x1, int j )
{
    float k = 1 ;
    int m ;
    for ( m = 1 ; m <= j ; m++ )
        k *= x1 ;
    return ( k ) ;
}

/* Calculate factorial */
float denominator ( int j )
{
    int m ;
    float h = 1 ;
    for ( m = 1 ; m <= j ; m++ )
        h = h * m ;
    return ( h ) ;
```



```
}
```

[24] Given three variables **x, y, z** write a function to circularly shift their values to right. In other words if x = 5, y = 8, z = 10 after circular shift y = 5, z = 8, x =10 after circular shift y = 5, z = 8 and x = 10. Call the with variables **a, b, c** to circularly shift values.

Program:

```
/* Circular shifting of values */

#include <stdio.h>
#include <conio.h>

main( )
{
    int x, y, z ;
    clrscr( ) ;
    printf ( "Enter the values x, y, z:\n" ) ;
    scanf ( "%d%d%d", &x, &y, &z ) ;
    printf ( "\nValues of x, y & z as entered:" ) ;
    printf ( "\nx = %d\ty = %d\tz = %d", x, y, z ) ;
    fun ( x, y, z ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* Function to shift values of x, y & z */
fun ( int x, int y, int z )
{
    int i, t ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        t = z ;
        z = y ;
```

```
        y = x ;
        x = t ;
        printf ( "\n\nAfter right shifting of values %d time(s):\n", i+1 ) ;
        printf ( "x = %d\ty = %d\tz = %d", x, y, z ) ;
    }
}
```

[25] Write a function to find the binary equivalent of a given decimal integer and display it.

Program:

```
/* Binary equivalent of a decimal number */

#include <stdio.h>
#include <conio.h>

main( )
{
    int num ;
    clrscr( ) ;
    printf ( "\nEnter the number: " ) ;
    scanf ( "%d", &num ) ;

    binary ( num ) ; /* Function call */

    printf ( "\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* function to convert decimal to binary */
binary ( int n )
{
    int r ;
    r = n % 2 ;
    n = n / 2 ;
    if ( n == 0 )
```



```
    {
        printf ( "\nThe binary equivalent is %d", r ) ;
        return ( r ) ;
    }
    else
        binary ( n ) ; /* Recursive call */
        printf ( "%d", r ) ;
}
```

[26] Given **N** and **K**. Write a function to compute the binomial coefficient

$$\frac{N!}{K!(N-K)!}$$

Program:

```
/* Compute binomial coefficient */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    long int n, k, z ;
    long int fact ( long int i ) ;

    clrscr( ) ;

    printf ( "\nEnter the two positive integers n and k where n > k: " ) ;
    scanf ( "%ld%ld", &n, &k ) ;
    z = fact ( n ) / ( fact ( k ) * fact ( abs ( n - k ) ) ) ;

    printf ( "Binomial Coefficient is %ld", z ) ;
```

```
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* Calculate factorial */
long int  fact ( long int  i )
{
    int  p = 1 ;
    while ( i != 0 )
    {
        p = p * i ;
        i-- ;
    }
    return ( p ) ;
}
```

[27] If the lengths of the sides of a triangle are denoted by **a**, **b**, and **c**, then area of triangle is given by

$$area = \sqrt{S(S-a)(S-b)(S-c)}$$

where, S = (a + b + c) / 2

Program:

```
/* Area of triangle with sides a, b & c */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    float  a, b, c, z ;
    float  area ( float  a, float  b, float  c ) ;
```



```
    clrscr( ) ;

    printf ( "\nEnter three sides of the triangle: " ) ;
    scanf ( "%f%f%f", &a, &b, &c ) ;
    z = area ( a, b, c ) ;

    printf ( "\n\nArea of the triangle = %.3f", z ) ;
    getch( ) ;
}

/* Function to calculate area from a formula */
float  area ( float  a, float  b, float  c )
{
    float  s, m, x ;
    s = ( a + b + c ) / 2 ;
    m = s * ( s - a ) * ( s - b ) * ( s - c ) ;
    x = sqrt ( m ) ;
    return ( x ) ;
}
```

[28] Write a function to compute the distance between two points and use it to develop another function that will compute the area of the triangle whose vertices are **A(x1, y1)**, **B(x2, y2)**, and **C(x3, y3)**. Use these function to develop a function which returns a value 1 if the point **(x, y)** lines inside the triangle ABC, otherwise a value 0.

Program:

```
/* Calculate distance between two points and use the function
   to calculate area of triangle given its three vertices */

#include <stdio.h>
#include <conio.h>
#include <math.h>
```


www.indianfun.org

```
main( )
{
    int x1, x2, y1, y2 ;
    float z, x ;
    float distance ( int x1, int y1, int x2, int y2 ) ;
    float area( ) ;

    clrscr( ) ;

    printf ( "\nEnter the co-ordinates of two points: " ) ;
    scanf ( "%d%d%d%d", &x1, &y1, &x2, &y2 ) ;
    z = distance ( x1, y1, x2, y2 ) ;
    printf ( "\nDistance between Two points = %f", z ) ;
    getch( ) ;

    x = area( ) ;
    printf ( "\nArea of the triangle = %f", x ) ;
    getch( ) ;
}

/* Function to calculate distance */
float distance ( int x1, int y1, int x2, int y2 )
{
    float m, d ;
    m = pow ( ( x2 - x1 ), 2 ) + pow ( ( y2 - y1 ), 2 ) ;
    d = sqrt ( m ) ;
    return d ;
}

/* Get vertices of triangle */
float area( )
{
    float triarea (float , float , float ) ;
    float a, b, c, d ;
    int x1, x2, x3, x4, y1, y2, y3, y4 ;
    float area1, area2, area3, totarea ;
    float a1, b1, c1 ;
```



```
    printf ("\nEnter the co-ordinates of three points: " ) ;
    scanf ("%d%d%d%d%d%d", &x1, &y1, &x2, &y2, &x3, &y3 ) ;
    printf ("\nEnter the co-ordinates of one more point: " ) ;
    scanf ( "%d%d", &x4 , &y4 ) ;

    a = distance ( x1, y1, x2, y2 ) ;
    b = distance ( x1, y1, x3, y3 ) ;
    c = distance ( x2, y2, x3, y3 ) ;

    d = triarea ( a, b, c ) ;
    a1 = distance ( x1, y1, x4, y4 ) ;
    b1 = distance ( x2, y2, x4, y4 ) ;
    c1 = distance ( x3, y3, x4, y4 ) ;

    area1 = triarea ( a, a1, b1 ) ;
    area2 = triarea ( b, a1, c1 ) ;
    area3 = triarea ( c, b1, c1 ) ;
    totarea = area1 + area2 + area3 ;

    if ( ( totarea - d ) <= 0.0009 )
    /* 0.0009 used to counter the anomaly in floating point
      calculations */
        printf ( "\nPoint ( %d, %d ) is inside the Triangle", x4, y4 ) ;
    else
        printf ( "\nPoint ( %d, %d ) lies outside the Triangle", x4, y4 ) ;

    return d ;
}

/* Function to calculate area from a formula */
float triarea ( float a, float b, float c )
{
    float s, m, x ;
    s = ( a + b + c ) / 2 ;
    m = s * ( s - a ) * ( s - b ) * ( s - c ) ;
    x = sqrt ( m ) ;
    return ( x ) ;
}
```

[29] Using the formulae

$$r = \sqrt{x^2 + y^2 + z^2}$$
$$\theta = \tan^{-1}(\sqrt{x^2 + y^2} / Z)$$
$$\varphi = \tan^{-1}(y/x)$$

write a function that will transform cartesian coordinates into spherical coordinates.

Program:

```
/* Convert cartesian co-ordinates to spherical */

#include <stdio.h>
#include <math.h>
#define PI 3.1415922

main( )
{
    /* Function prototypes */
    float  spherical_r ( float, float, float  ) ;
    float  spherical_O ( float, float, float  ) ;
    float  spherical_Q ( float, float  ) ;
    float  x, y, z ;
    float  r, O, Q ;

    printf ( "\n Enter the X, Y, Z coordinates: " ) ;
    scanf ( "%f%f%f", &x, &y, &z ) ;

    r = spherical_r ( x, y, z ) ;
    printf ( "\n Spherical Co-ordinates are: \n" ) ;
    printf ( "\n\tr = %f Units", r ) ;

    Q = spherical_Q ( x, y ) ;
    printf ( "\n\tQ = %f degrees", Q ) ;
```



```
    O = spherical_O ( x, y, z ) ;
    printf ( "\n\tO = %f degrees ", O ) ;

    getch( ) ;
}

/* Function to calculate radial distance between point  & origin */
float  spherical_r ( float  x, float  y, float  z )
{
    float  a ;
    a = sqrt ( x * x + y * y + z * z ) ;
    return a ;
}

/* Function to calculate angle between x,y plane & point  */
float  spherical_O ( float  x, float  y, float  z )
{
    float  a , t ;
    if ( z == 0 )
        a = 0 ;
    else
        a = ( atan ( sqrt ( x * x + y * y ) / z ) ) * ( 180.0/PI )

    return a ;
}

/* Funtion to calculate angle between x axis & the point  */
float  spherical_Q ( float  x, float  y )
{
    float  a ;

    if ( x == 0 && y == 0 )
        a = 0 ;
    else
        a = ( atan ( y / x ) ) * ( 180 / PI ) ;

    return a ;
```

}

[30] Write another function to calculate **N**! Which is define as 1.2.3.4....N, and 0! Is defined to be 1.

Write another function to calculate the Binomial Coefficient given by.

$$\frac{N!}{K!(N-K)!}$$

Then write a **main( )** that makes use of the aforesaid functions to evaluate:

(a) $p = \frac{(i+j+k)!}{i!\ j!\ k!}$

where **p** is the number of permutations on a set of **m** objects which **i** are of one type, **j** are of a second type, and **k** are of a third type, it being assumed that object of any of the three given types are indistinguishable from one another.

(b) $PROB = \frac{N!}{K!(N-K)!} p^{k} q^{n-k}$

Where in a sequence of independent trials of an experiment each trial has a probability of success **p** and a probability of failure **q**, and PROB is the probability of exactly **k** successes in **n** trials of the experiment.

Program:

```
#include <stdio.h>
#include <conio.h>

/* Function to calculate factorial */
float fact ( float i )
{
   float p = 1 ;
   while ( i != 0 )
   {
        p = p * i ;
        i-- ;
   }
   return p ;
}

/* Function to calculate binomial coefficient */
float binomial ( float n , float k )
{
   float z ;
   z = fact ( n ) / ( fact ( k ) * fact ( abs ( n - k ) ) ) ;
   return z ;
}
main( )
{
   float i, j, k, p, prob, K, N, n, q ;
   clrscr( ) ;
   printf ( "Enter the three numbers i, j and k where i > j > k:\n" ) ;
   scanf ( "%f%f%f", &i, &j, &k ) ;

   p = fact (i + j + k ) / ( fact ( i ) * fact ( j ) * fact ( k ) ) ;
   printf ( "\np = %f\n", p ) ;

   printf ( "Enter the values of N and K: " ) ;
   scanf ( "%f%f", &N, &K ) ;

   printf ( "Enter the values of 'n' and 'q': " ) ;
   scanf ( "%f%f", &n, &q ) ;

   prob = binomial ( N, K ) * pow ( p, K ) * pow ( q, ( n - k ) ) ;
   printf ( "PROB = %f", prob ) ;

   getch( ) ;
```

```
}
```

[31] Write a function to compute the greatest common divisor given by Euclid's algorithm, exemplified for J = 1980, K = 1617 as follows:

1980/1617 = 1	1980 – 1 * 1617 = 363
1617/363 = 4	1617 – 4 * 363 = 165
363/165 = 2	363 – 2 * 165 = 33
5/33 = 5	165 – 5 * 33 = 0

Thus, the greatest common divisor is 33.

Program:

```
/* Compute Greatest common diviser - Euclid's Algoritm */

#include <stdio.h>
#include <conio.h>

main( )
{
    int j, k, t, r, z ;
    clrscr( ) ;
    printf ( "\nEnter two numbers:" ) ;
    scanf ( "%d%d", &j, &k ) ;
    z = fun ( j, k ) ;
    printf ( "\nThe greatest common diviser of the two numbers is %d" ,
              z ) ;
    getch( ) ;
}

/* Function to calculate GCD */
fun ( int j, int k )
{
    int r = 1, m, n = 0, t ;
    if ( k > j ) /* Interchange values */
    {
        t = j ;
        j = k ;
        k = t ;
    }
    else
    {
        if ( j == k )
            return j ;
    }
    while ( 1 )
    {
        r = j / k ;
        m = j - ( r * k ) ;

        if ( ! ( j % k ) ) /* j is exact multiple of k */
          n = k ;

        if ( m == 0 )
            break ;

        j = k ;
        k = m ;
        n = m ;
    }
    return n ;
}
```


Pointers

[32] Given the following information Answer the questions which follow

Variable Name	Address	Contents
P	2568	425
Q	4284	2568
R	6242	4284
A[0]	8468	232

www.indianfun.org

A[10]	8488	2568

Answer

(a)	&p = ?	2568
(b)	*p = ?	425
(c)	**r = ?	425
(d)	&q = ?	4284
(e)	*(&q) = ?	2568
(f)	&(*r) = ?	4268
(g)	&a[0] = ?	8468
(h)	a + ?	8468
(i)	a[0] = ?	232
(j)	&a[5] = ?	8488
(k)	a[10] = ?	2568
(l)	*a[10] = ?	425

[33] Match the following with reference to the following program segment:

```
int i, j, = 25;
int  *pi, *pj = & j;
.......
....... /* more lines of program */
.........
*pj = j + 5;
j = *pj + 5 ;
pj = pj ;
*pi = i + j
```

Each integer quantity occupies 2 bytes of memory. The value assigned to I begin at (hexadecimal) address F9C and the value assigned to j begins at address F9E. Match the value represented by Left Hand Side quantities with the Right.

1.	&i	a.	30
2.	&j	b.	F9E
3.	Pj	c.	35
4.	*pj	d.	FA2
5.	I	e.	F9C
6.	pi	f.	67
7.	*pi	g.	unspecified
8.	(pi + 2)	h.	65
9.	(*pi + 2)	i.	F9E
10.	* (pi + 2)	j.	F9E
		k.	FAO
		l.	F9D

Answer

1.	&i	e.	F9C
2.	&j	b.	F9E
3.	Pj	i.	F9E
4.	*pj	h.	65
5.	i	c.	35
6.	Pi	j.	F9E
7.	*pi	h.	65
8.	(pi + 2)	d.	FA2
9.	(*pi + 2)	f.	67
10.	* (pi + 2)	g.	unspecified

[34] Match the following with reference to the following segment:

```
int X[3][5] = {
                {1, 2, 3, 4, 5 },
                {6, 7, 8, 9, 10},
                {11, 12, 13, 14, 15 }
              }, *n = &x ;
```

1.	*(**x + 2) + 1)	a.	9
2.	*(*x + 2) + 5	b.	13
3.	*(*(x + 1))	c.	4
4.	*(*(x) + 2) + 1	d.	3
5.	* (*(x + 1) + 3)	e.	2
6.	*n	f.	12
7.	*(n +2)	g.	14
8.	(*(n + 3) + 1	h.	7

9.	*(n + 5)+1	i.	1
10.	++*n	j.	8
		k.	5
		l.	10
		m.	6

Answer:

1.	*(**x + 2) + 1)	f.	12
2.	*(*x + 2) + 5	j.	8
3.	*(*(x + 1))	m.	6
4.	*(*(x) + 2) + 1	c.	4
5.	* (*(x + 1) + 3)	a.	9
6.	*n	i.	1
7.	*(n +2)	d.	3
8.	(*(n + 3) + 1	k.	5
9.	*(n + 5)+1	h.	7
10.	++*n	e.	2

[35] Match the following with reference to the following program segment:

```
struct
{
    int x, y;
} s[ ] = { 10, 20, 15, 25, 8, 75, 6, 2 };
int *i ;
i = s ;
```

1.	*(i + 3)	a.	85
2.	s[i[7]].x	b.	2
3.	s[(s + 2)->y / 3[I]]-y	c.	6
4.	i[i[1]-i[2]]	d.	7
5.	i[s[3].y]	e.	6
6.	(s + 1)->x + 5	f.	15
7.	(1 +i)*(i + 4) /*i	g	25
8.	s[i[0] – i[4]].y + 10	h.	8
9.	(*(s + *(i + 1)/ * i)). x + 2	i.	1

10.	++i[i[6]]	j.	100
		k.	10
		l.	20

Answer:

1.	*(i + 3)	g.	25
2.	S[i[7]].x	h.	8
3.	S[(s + 2)->y / 3[I]]-y	b.	2
4.	i[i[1]-i[2]]	i.	75
5.	i[s[3].y]	f.	15
6.	(s + 1)->x + 5	l.	20
7.	(1 +i)*(i + 4) /*i	e.	16
8.	S[i[0] – i[4]].y + 10	a.	85
9.	(*(s + *(i + 1)/ * i)). x + 2	k.	10
10.	++i[i[6]]	d.	7

[36] Match the following with reference to the following program segment: segment

```
unsigned int arr[3][3] = {
                          2, 4, 6,
                          9, 1, 10,
                          16, 64, 5
                        };
```

1.	**arr	a.	64
2.	**arr < *(*arr + 2)	b.	18
3.	*(arr + 2) / (*(*arr + 1) > **arr)	c.	6
4.	*(arr[1] + 1) | arr[1][2]	d.	3
5.	*(arr[0]) | *(arr[2])	e.	0
6.	arr[1][1] < arr[0][1]	f.	16
7.	arr[2][[1] & arr[2][0]	g.	1
8.	arr[2][2] | arr[0][1]	h.	11
9.	arr[0][1] ^ arr[0][2]	i.	20
10.	++**arr + --arr[1][1]	j.	2
		k.	5
		l.	4

Answer:

1.	**arr	j. 2
2.	**arr < *(*arr + 2)	g. 1
3.	*(arr + 2) / (*(*arr + 1) > **arr)	c. 6
4.	*(arr[1] + 1) | arr[1][2]	h. 11
5.	*(arr[0]) | *(arr[2])	b. 18
6.	arr[1][1] < arr[0][1]	g. 1
7.	arr[2][1] & arr[2][0]	e. 0
8.	arr[2][2] | arr[0][1]	k. 5
9.	arr[0][1] ^ arr[0][2]	j. 2
10.	++**arr + --arr[1][1]	d. 3

Arrays and Strings

[37] Write a program to find the equation of a straight line passing through n pairs of x and y coordinates.

Program:

```
/* Equation of a straight line from n pair of co-ordinates */

#include <stdio.h>
#include <conio.h>

main( )
{
    int  x[3], y[3], i ;
    float  m, m1, m2 ;
    clrscr( ) ;
    printf ( "Enter the co-ordinates" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        scanf ( "%d", &x[i] ) ;
        scanf ( "%d", &y[i] ) ;
    }
    /* Calculate slope of the line */
```



```
    m = ( y[1] - y[0] ) * 1.0 / ( x[1] - x[0] ) ;
    for ( i = 1 ; i <= 2 ; i++ )
    {
        m1 = ( y[i] - y[0] ) * 1.0 / ( x[i] - x[0] ) ;
        if ( m == m1 )
            continue ;
        else
        {
            printf ( "Not a stright line" ) ;
            break ;
        }
    }
    if ( i == 3 )
    {
        printf ( "It is a stright line" ) ;
        printf ( "\nEquation of this line is" ) ;
        m1 = y[0]*1.0 - m * x[ 0] ;

        if (m == 0.0 )
            m1 = m ;
        printf ( "\nY = %.2f * X + %f ", m, m1 ) ;
    }
    getch( ) ;
}
```

[38] Write a program which interchanges the odd and even components of an array.

Program:

```
/* Interchanging odd & even components of an array */

#include <stdio.h>
#include <conio.h>

main( )
{
```


www.indianfun.org

```
    int a[ ] = { 1,2,3,4,5,6,7,8,9,10 } ;
    int i, j ;

    clrscr( ) ;

    printf ( "\nOriginal Array:\n" ) ;
    for ( i = 0 ; i <= 9 ; i++ )
        printf ( "%d ", a[i] ) ;

    for (i = 0 ; i <= 9 ; i += 2 )
    {
        for ( j = i + 1 ; j<= 9 ; j++ )
            swap ( &a[i], &a[i+1] ) ;

    }
    printf ( "\n\nModified Array:\n" ) ;
    for ( i = 0 ; i <= 9 ; i++ )
        printf ( "%d ", a[i] ) ;

    getch( ) ;
}

/* Function to swap elements */
swap ( int *i, int *j )
{
    int t ;
    t = *i ;
    *i = *j ;
    *j = t ;
}
```

[39] Write a program to find if a square matrix is symmetric.

Program:

```
/* To check if a square matrix is symmetric */
```



```
#include <stdio.h>
#include <conio.h>

main( )
{
    int arr[3][3], i, j, count = 0 ;
    clrscr( ) ;
    printf ( "\nEnter the elements of the Matrix:\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
            scanf ( "%d", &arr[i][j] ) ;
    }

    /* Print the matrix as entered */
    printf ( "\nThe matrix formed is....\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
            printf ( "%d ", arr[i][j] ) ;
            printf ( "\n" ) ;
    }

    /* Check for symmetry */
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = i ; j <= 2 ; j++ )
        {
            if ( arr[i][j] == arr[j][i] )
                count++ ;
        }
    }

    if ( count == 6 )
        printf ( "\n\nThe matrix is symmetric" ) ;
    else
        printf ( "\n\nThe matrix is not symmetric" ) ;
```

```
    getch( ) ;
}
```

[40] A factory has 3 division and stocks 4 categories of products. An inventory table is updated for each division and for each produce as they are received. There are three independent suppliers of products to the factory:

(a) Design a data format to represent each transaction.
(b) Write a program to take a transaction and update the inventory.
(c) Id the cost per item is also given write a program to calculate the total inventory values.

Program:

```
/* Inventory Management */

#include <stdio.h>
#include <conio.h>

main( )
{
    int *inven[3][4] ; /* 3 divisions and 4 categories */
    int comp[3] ;
    int *c ;
    char ch = 'y' ;
    int t, i, j, pno, d, comp_no ;
    int tot = 0, cost[4] = { 100, 200, 300, 400 } ;
    long int tot_in_val = 0 ;

    clrscr( ) ;

    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 3 ; j++ )
        {
            c = ( int* ) malloc ( sizeof ( int ) * 3 ) ;
```



```
            inven[i][j] = c ;
            *( inven[i][j] + 0 ) = 0 ;
            *( inven[i][j] + 1 ) = 0 ;
            *( inven[i][j] + 2 ) = 0 ;
        }
    }

    clrscr( ) ;
    while ( ch == 'y' || ch == 'Y' )
    {
        clrscr( ) ;
        printf ( "\nEnter the Division 0, 1 or 2: " ) ;
        scanf ( "%d", &d ) ;
        if ( d < 0 || d > 2 )
        {
            printf ( "\nInvalid Division" ) ;
            continue ;
        }

        printf ( "\nSelect product number\nfrom 0, 1, 2 or 3 which
                represents its category:" ) ;
        scanf ( "%d", &pno ) ;
        if ( pno < 0 || pno > 3 )
        {
            printf ( "\nInvalid Division" ) ;
            continue ;
        }

        printf ( "\nEnter the company number 0, 1, 2\nfrom where the
                product is purchased:" ) ;
        scanf ( "%d", &comp_no ) ;

        if ( comp_no >= 0 && comp_no <= 2 )*( inven[d][pno] +
            comp_no ) = *( inven[d][pno] + comp_no ) + 1 ;
        else
        {
            printf ( "\nInvalid Company number " ) ;
            continue ;
```

```
    }
    clrscr( ) ;
    printf ( "\n\n\t\t Want to enter another data ( y / n )\n\t\t " ) ;
    fflush ( stdin ) ;
    scanf ( "%c", &ch ) ;
    fflush ( stdin ) ;
}

clrscr( ) ;
printf ("\ntotal quantity of products present in each divisions are:\n") ;
printf ( "\nProduct \t0\t1\t2\t3\n" ) ;
for ( i = 0 ; i <= 2 ; i++ )
{
    printf ( "\nDivision No %d", i ) ;
    for ( j = 0 ; j <= 3 ; j++ )
    {
        t = *( inven[i][j] + 0 ) + *( inven[i][j] + 1 ) + *( inven[i][j] + 2 ) ;
        printf ( "\t%d", t ) ;
    }
    printf ( "\n" ) ;
}
getch( ) ;
clrscr( ) ;
printf ( "\nTotal cost of products present in each divisions are:\n" ) ;

for ( i = 0 ; i <= 2 ; i++ )
{
    printf ( "\nDivision No %d", i+1 ) ;
    for ( j = 0 ; j <= 3 ; j++ )
    {
        t = *( inven[i][j] + 0 ) + *( inven[i][j] + 1 ) + * ( inven[i][j] + 2 ) ;
        tot = tot + t * cost[j] ;
    }
    printf ( "\tRs. %d ", tot ) ;
    printf ( "\n" ) ;
    tot_in_val = tot_in_val + tot ;
    tot = 0 ;
}
```



```
    printf ( "\n\n Total inventory cost = %ld", tot_in_val ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

[41] Write a function to find the norm of a matrix. The norm is defined as the square root of the sum of squares of all elements in the matrix.

Program:

```
/* To find the norm of a Matrix */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    int  matrix[3][3] ;  /* 3 x 3 matrix */
    int  i, j ;
    int  sum = 0 ;
    float  norm ;
    clrscr( ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        printf ( "\nEnter the elements of the %d row\n", i ) ;
        for ( j = 0 ; j <= 2 ; j++ )
        {
            scanf ( "%d", &matrix[i][j] ) ;
        }
    }

    /* Calculate the norm */
    for ( i = 0 ; i <= 2 ; i++ )
    {
```

```
        for ( j = 0 ; j <= 2 ; j++ )
        {
            sum = sum + pow ( matrix[i][j], 2 ) ;
        }
    }
    clrscr( ) ;

    /* PRint the matrix as entered */
    printf ( "\nYou entered the matrix as :\n" ) ;
    for ( i = 0 ; i <= 2 ; i++ )
    {
        for ( j = 0 ; j <= 2 ; j++ )
        {
            printf ( "\t%d", matrix[i][j] ) ;
        }
        printf ( "\n" ) ;
    }

    norm = sqrt ( sum ) ;
    printf ( "\n\nNorm of this matrix = %f", norm ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

[42] A dequeue is an ordered set of elements in which elements may be inserted or retrieved from either end. Using an array simulate a dequeue of characters and the operations retrieve left, retrieve right, insert left, insert right. Exceptional conditions such as dequeue full or empty should be indicated. Two pointers (namely, left and right) are needed in this simulation.

Program:

```
/* Implementation of a deque using an array */
```



```
#include <stdio.h>
#include <conio.h>

/* Function prototypes */
void add_front ( int ) ;
void add_rear ( int ) ;
int  retrieve_front ( void ) ;
int  retrieve_rear ( void ) ;
void display ( void ) ;

/* Global variables */
int  *front, *rear ;
int  a[26] ;

main( )
{
    int  item ;
    front = NULL ;
    rear = NULL ;

    clrscr( ) ;
    printf ( "\nAdding elements at front of a deque: " ) ;
    add_front ( 10 ) ;
    add_front ( 40 ) ;
    add_front ( 30 ) ;
    display( ) ;
    getch( ) ;

    printf ( "\n\nAdding an element at the rear of a deque: " ) ;
    add_rear ( 50 ) ;
    display( ) ;

    printf ( "\n\nRetreiving an element from the front of a deque: " ) ;
    item = retrieve_front( ) ;
    if ( item == -1 )
        printf ( "\nDeQueue Empty " ) ;
    else
        printf ( "\n\nFront Item = %d ", item ) ;
```

```
    display( ) ;

    printf ( "\n\nRetreiving an element from the rear of a deque: " ) ;
    item = retrieve_rear( ) ;
    if ( item == -1 )
        printf ( "\nDeQueue Empty " ) ;
    else
        printf ( "\n\nRear Item = %d ", item ) ;
    display( ) ;

    getch( ) ;
}

/* Function to retrieve item from rear */
int  retrieve_rear ( void )
{
    int  item, i, j ;
    if ( rear == NULL )
    {
        printf ( "\n DeQueue Empty " ) ;
        return -1 ;
    }
    else
    {
        item = *rear ;
        i = rear - front ;
        if ( i == 0 )
        {
            front = NULL ;
            rear = NULL ;
        }
        else
        {
            *rear = 0 ;
            rear-- ;
        }
    }
    return item ;
```



```
}

/* Function to retrieve item from front */
int  retrieve_front ( void )
{
    int  item, i, j ;
    if ( front == NULL )
    {
        printf ( "\n DeQueue Empty " ) ;
        return -1 ;
    }
    else
    {
        item = *front ;
        i = rear - front ;
        if ( i == 0 )
        {
            front = NULL ;
            rear = NULL ;
        }
        else
        {
            for ( j = 0 ; j <= i - 1 ; j ++ )
                front [ j ] = front [ j + 1 ] ;
            *rear = 0 ;
            rear-- ;
        }
    }
    return item ;
}

/* Function to add item to rear */
void add_rear ( int  item )
{
    int  i, j ;
    if ( front == NULL )
    {
        front = a ;
```

```
            *front = item ;
            rear = a ;
        }
        else
        {
            i = rear - front ;

            if ( i != 25 )
            {
                rear++ ;
                *rear = item ;
            }
            else
                printf ( "\nDeQueue full " ) ;
        }
}

/* Function to add item at front */
void add_front ( int  item )
{
    int  i, j ;
    if ( front == NULL )
    {
        front = a ;
        *( front ) = item ;
        rear = a ;
    }
    else
    {
        front = a ;
        j = rear - front ;
        if ( j != 25 )
        {
            for ( i = j + 1 ; i >= 0 ; i-- )
                front [ i ] = front [ i - 1 ] ;
            *( front + 0 ) = item ;
            rear++ ;
        }
```



```
        else
            printf ( "\n DeQueue Full " ) ;
    }
}

/* Function to display the deque */
void display ( void )
{
    int  i ;
    int  *p = front ;
    if ( p == NULL )
    {
        printf ( "\n\nDeQueue Empty " ) ;
    }
    else
    {
        printf ( "\n" ) ;
        for ( i = 0 ; i <= ( rear - front ) ; i++ )
            printf ( "\n a[%d] = %d ", i, *p++ ) ;
    }
}
```

[43] Write a program which concatenates a string to the left of a given string.

Program:

```
/* Concatenation of strings */

#include <stdio.h>
#include <conio.h>

main( )
{
    char  str1[10], str2[10] ;
    int  t ;
```

```
    clrscr( ) ;

    printf ( "\nEnter the two strings: " ) ;
    scanf ( "%s%s", str1, str2 ) ;
    printf ( "\nString1 = %s\tString2 = %s ", str1, str2 ) ;

    /* Add two strings */
    preconcatenate ( str2, str1 ) ;
    printf ( "\n\nAfter adding String2 to the left of String1:" ) ;
    printf ( "\nThe new string is: %s", str2 ) ;
    getch( ) ;
}

/* Function to add string at the left of a given string */
preconcatenate ( char *t, char *s )
{
    while ( *t )
        t++ ;
    while ( *s )
        *t++ = *s++ ;
    *t = '\0' ;/* String termination character */
}
```

[44] Write a program to delete all vowels from a sentence. Assume that the sentence is not more than 80 characters long.

Program:

```
/* Delete all vowels from a sentence */

#include <stdio.h>
#include <conio.h>

main( )
{
    char str[80], str1[80] ;
    char *s, *p ;
```



```
    clrscr( ) ;
    printf ( "\nEnter a sentence not more than 80 characters long:\n" ) ;
    gets( str ) ;
    s = str ; /* Base address of the string */
    p = str1 ; /* Base address of new string */
    while ( *s )
    {
        if ( *s == 'a' || *s == 'e' || *s == 'i' || *s == 'o' || *s == 'u' )
            s++ ;
        else
        if ( *s == 'A' || *s == 'E' || *s == 'I' || *s == 'O' || *s == 'U' )
            s++ ;
        else
            *p++ = *s++ ;
    }
    *p = '\0' ;
    printf ( "\n\nSentence after removing all vowels is:\n" ) ;
    puts ( str1 ) ;
    getch( ) ;
}
```

[45] Write a program which will read a line and delete from it all occurrences of the word 'the'.

Program:

```
/* Delete all occurrences of "the" from a sentence */

#include <stdio.h>
#include <conio.h>

main( )
{
    char str[80], str2[80] ;
    char *s, *q, *p ;
    int i ;
    clrscr( ) ;
```


www.indianfun.org

```
printf ( "\nEnter a sentence not more than 80 chars long:\n" ) ;
gets ( str ) ;
s = str ; /* Base address of the string */
p = str2 ; /* Base address of new string */
while ( *s )
{
    q = s ;
    if ( *s == 't' || *s == 'T' )
    {
        s++ ;
        if ( *s == 'h' )
        {
            s++ ;
            if ( *s == 'e' )
            ;
            else
            {
                for ( i = 0; i <= 2 ; i++ )
                    *p++ = *q++ ;
            }
        }
        else
        {
            *p++ = *q++ ;
            s-- ;
        }
    }
    else
        *p++ = *s ;
    s++ ;
}
*p = '\0' ;
printf ( "\n\nSentence after deleting all occurences of 'the' is:\n" ) ;
puts ( str2 ) ;
getch( ) ;
}
```


[46] Write a program which takes a set of names of individuals and abbreviates the first, middle and other names except the last name by their first letter.

Program:

```
/* Convert a full name into initials & lastname */

#include <stdio.h>
#include <conio.h>

main( )
{
    char  str[30] ;
    char  *p ;
    int  count = 0, l ;
    clrscr( ) ;
    printf ( "\nEnter name, middle name and surname:\n" ) ;
    gets ( str ) ;
    p = str ;/* Base address of the string */

    /* Count number of blank spaces */
    while ( *p )
    {
        if ( *p == ' ' )
            count++ ;
        p++ ;
    }
    p = str ;

    printf ( "\nThe name converted to initials is: " ) ;
    while ( count )
    {
        printf ( "%c.", *p ) ;
        while ( *p != ' ')
            p++ ;
        p++ ;
```

```
        count-- ;
    }
    printf ( "%s", p ) ;
    getch( ) ;
}
```

[47] Write a program to count the number of occurrences of any two vowels in succession in a line of text. For example, in the following sentence:

"Pleases read this application and give me gratuity"

Such occurrences are ea, ea, ui.

Program:

```
/* Check for 2 vowels in succession */

#include <stdio.h>
#include <conio.h>

main( )
{
    char  str[80] ;
    int  count = 0 ;
    char  *s = str ;

    clrscr( ) ;
    printf ( "\nEnter the string:\n" ) ;
    gets ( str ) ;

    while ( *s )
    {
        if ( *s == 'a' || *s == 'e' || *s == 'i' || *s == 'o' || *s == 'u' )
        {
            s++ ;
```



```
            if ( *s == 'a' || *s == 'e' || *s == 'i' || *s == 'o' || *s == 'u' )
                count ++ ;
        }
        s++ ;
    }
    printf ( "The number of occurrences of two successive vowels: %d" ,
            count ) ;
    getch( ) ;
}
```

[48] A queue is a data structure in which a new element is inserted at the back of the queue and an element is retrieved from the front (the other end) of the queue. For example, given a queue,

```
  A B C D E F X Y Z
front              back
```

A retrieval operation will retrieve **A**. An insert operation will insert an element behind **Z**. Write functions to add an element to a queue and to retrieve an element from the queue. The functions must have parameters to indicate queue full, queue empty conditions and set point for adding and retrieving elements.

Program:

```
/* Implementation of a queue using array */

#include <stdio.h>
#include <conio.h>

/* Global variable */
char  queue[26] ;
int  i ;
```


www.indianfun.org

```
/* Function prototypes */
char  retrieve( void ) ;
void add ( char  c ) ;
void display ( void ) ;

main( )
{
    char  item ;
    i = -1 ;  /* indicates empty queue */
    clrscr( ) ;
    /* Add elements to a queue */
    printf ( "\nElements of Queue are : \n" ) ;
    add ( 'a' ) ;
    add ( 'b' ) ;
    add ( 'c' ) ;
    add ( 'd' ) ;
    display( ) ;
    getch( ) ;

    /* Reetreive elements from queue */
    item = retrieve( ) ;
    printf ( "\n\nAfter retreiving element = %c", item ) ;

    printf ( "\nElements of Queue are : \n" ) ;
    display( ) ;
    getch( ) ;

    item = retrieve( ) ;
    printf ( "\n\nAfter retreiving element = %c", item ) ;
    getch( ) ;

    printf ( "\nElements of Queue are : \n" ) ;
    display( ) ;
    getch( ) ;
}

/* Function to add elements to a queue */
void add ( char  c )
```



```
{
    if ( i == 25 )
    {
        printf ( "\nQueue Full " ) ;
        return ;
    }
    else
    {
        i++ ;
        queue[i] = c ;
    }
}

/* Function to display the queue */
void display ( void )
{
    int  j ;
    if ( i == -1 )
    {
        printf ( "\n Queue empty " ) ;
        return ;
    }
    for ( j = 0 ; j <= i ; j++ )
        printf ( "%c\t", queue[ j ] ) ;
    return ;
}

/* Function to retrieve elements from a queue */
char  retrieve ( void )
{
    char  item ;
    int  j ;
    if ( i == -1 )
    {
        printf ( "\nQueue Empty " ) ;
        return 2;
    }
    else
```

```
    {
        item = queue[0] ;
        for ( j = 0 ; j <= i - 1 ; j++ )
        {
            queue[j] = queue[j + 1 ] ;
        }
        i-- ;
        return item ;
    }
}
```

[49] Given an array **p[5]**, write a function to shift it circularly left by two positions. Thus, if p[0] = 15, p[1]= 30, p[2] = 28, p[3]= 19, p[4] = 61 then after the shift p[0] = 28, p[1] = 19, p[2] = 61, p[3] = 15 and p[4] = 30. Call this function with a (4 x 5) matrix and get its rows left shifted.

Program:

```
/* Circular rotation of array elements and its use in matrix */

#include <stdio.h>
#include <conio.h>

/* Function prototype */
void swap ( int  *p ) ;

main( )
{
    int  p[4][5] ;
    int  i, j ;
    clrscr( ) ;
    /* Enter data for a matrix */
    for ( i = 0 ; i <= 3 ; i++ )
    {
        printf ( "\nEnter the %d row elements:\n", i ) ;
```



```
        for ( j = 0 ; j <= 4 ; j++ )
        {
            scanf ( "%d", &p[i][j] ) ;
        }
    }
    /* Print  matrix as entered */
    printf ( "\nMatrix elements as entered:" ) ;
    for ( i = 0 ; i <= 3 ; i++ )
    {
        printf ( "\nThe %d row elements:  ", i ) ;

        for ( j = 0 ; j <= 4 ; j++ )
        {
            printf ( "\t%d", p[i][j] ) ;
        }
    }
    getch( ) ;
    printf ( "\n" ) ;

    /* Function call to left shift rows */
    swap ( p ) ;  /* Base address of array passed to function */

    /* Print  new matrix after left shifting */
    printf ( "\nMatrix after left shifting the row elements:" ) ;
    for ( i = 0 ; i <= 3 ; i++ )
    {
        printf ( "\nThe %d row elements:  ", i ) ;

        for ( j = 0 ; j <= 4 ; j++ )
        {
            printf ( "\t%d", p[i][j] ) ;
        }
    }
    getch( ) ;
}

/* Function for left shifting of the rows */
void swap ( int  *p )
```

```
{
    int k, i, tt, t, j ;
    for ( k = 0 ; k <= 3 ; k++ ) /* 4 rows to be shifted */
    {
        for ( i = 0 ; i <= 1 ; i++ ) /* shifting done twice */
        {
            t = *( p + k * 5 + 0 ) ;
            for ( j = 0 ; j <= 3 ; j++ )
            {
                tt = *( p + k * 5 + j ) ;
                *( p + k * 5 + j ) = *( p + k * 5 + j + 1 ) ;
                *( p + k * 5 + j + 1 ) = tt ;
            }
            *( p + k * 5 + j ) = t ;
        }
    }
}
```

[50] A 6 x 6 matrix is entered through the keyboard and stored in a 2-dimensioned array **mat[7][7]**. Write a program to obtain the Determinant values of this matrix.

Program:

```
/* Determinant of a n x n matrix */

/* The logic used here is: to evaluate a n x n determinant we multiply
   the first row elements of this determinant with a corresponding n-1 x
   n-1 determinant by first copying the n-1 x n-1 determinant into pre-
   allocated memory. We continue this process recursively till we reach
   a 1 x 1 determinant and then retrace back the steps. */

#include <stdlib.h>
#include <alloc.h>
#include <conio.h>

main( )
```



```
{
    int *arr, sum, n, i, j, pos, num ;

    printf ( "\nEnter value of n for ( n x n ) matrix " ) ;
    scanf ( "%d", &n ) ;
    /* allocate memory to accommodate the determinant */
    arr = calloc ( n * n, 2 ) ;

    printf ( "\nEnter numbers :\n" ) ;
    for ( i = 0 ; i < n ; i++ )
    {
        for ( j = 0 ; j < n ; j++ )
        {
            scanf ( "%d", &num ) ;
            pos = i * n + j ;
            arr[pos] = num ;
        }
    }
    /* Print the matrix as entered */
    for ( i = 0 ; i < n ; i++ )
    {
        for ( j = i ; j < n*n ; j += n )
        {
            printf ( "%d\t", arr[ j ] ) ;
        }
    printf ( "\n" ) ;
    }

    sum = detmat ( arr, n ) ;
    free ( arr ) ; /* Memory used by arr is freed */

    printf ( "\n%d", sum ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* Function to calculate determinant */
```


www.indianfun.org

```
detmat ( int *arr, int order )
{
    int sign = 1, sum = 0, i, j, k, count, *arr2 ;
    int newsize, newpos, pos ;

    if ( order == 1 )
        return ( arr[0] ) ;

    for ( i = 0 ; i < order ; i++, sign *= -1 )
    {
    /* copy n-1 by n-1 array into another array */
    newsize = ( order - 1 ) * ( order - 1 ) ;
    arr2 = calloc ( newsize, 2 ) ; /* allocate memory for a new array */
        for ( j = 1 ; j < order ; j++ )
        {
            for ( k = 0, count = 0 ; k < order ; k++ )
            {
                if ( k == i )
                continue ;

                pos = j * order + k ;
                newpos = ( j - 1 ) * ( order - 1 ) + count ;

                arr2[newpos] = arr[pos] ;
                count++ ;
            }
        }

        /* find determinant value of n-1 by n-1 array and add it to sum */
        sum = sum + arr[i] * sign * detmat ( arr2, order - 1 ) ;
        free ( arr2 ) ; /* Free the memory used by the second array */
    }
    return ( sum ) ;
}
```

[51] For the following set of sample data, compute the standard deviation and the mean.

-6, -12, 8, 13, 11, 6, 7, 2, -6, -9, -10, 11, 10, 9, 2

The formula for standard deviation is

$$\frac{\sqrt{(x_i - \bar{x})^2}}{n}$$

Where **xi** are the data item and $\bar{x}$ is the mean.

Program:

```
/* Computation of standard deviation */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    int data[15] = {-6, -12, 8, 13, 11, 6, 7, 2, -6, -9, 2, 11, 10, 9, -10 } ;
    float xi, std[15], mean = 0 ;
    int i, n = 15 ;

    clrscr( ) ;

    /* print data items */
    for ( i = 0 ; i <= 14 ; i++ )
    {
        printf ( "%d\t", data[i] ) ;
    }

    /* Calculate mean for given data */
    for ( i = 0 ; i <= 14 ; i++ )
    {
        mean = mean + data[i] ;
    }
```


www.indianfun.org

```
    mean = mean / n ;

    /* Calculate standard deviation */
    for ( i = 0 ; i <= 14 ; i++ )
    {
        std[i] = sqrt ( pow ( ( data[i] - mean ), 2 ) ) / n ;
    }

    /* Print result */
    printf ( "\n\n Mean = %f\n\n", mean ) ;
    for ( i = 0 ; i <= 14 ; i++ )
    {
        printf ( "\nStandard Deviation of %d = %f", data[i], std[i] ) ;
    }
    getch( ) ;
}
```

[52] The area of a triangle can be computed by the sine low when 2 sides of the triangle and the angle between them are known.

Area = (1/2) an sin (angle)

Given the following 6 triangular pieces of land, write a program to find their area and determine which is largest,

Plot No.	A	B	angle
1	137.4	80.9	0.78
2	155.2	92.62	0.89
3	149.3	97.93	1.35
4	1600	100.25	9.00
5	155.6	68.95	1.25
6	149.7	120.0	1.75

Program:

```
/* Area = (1/2)ab sin ( angle ) */
```



```
#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    float t, a[6], b[6], angle[6], area[6], largest = 0.0 ;
    int i, plot ;
    clrscr( ) ;
    for ( i = 0 ; i <= 5 ; i++ )
    {
        printf ( "\nEnter the values of the following data:" ) ;
        printf ( "\na%d = ", i+1 ) ;
        scanf ( "%f", &a[i] ) ;
        printf ( "\nb%d = ", i+1 ) ;
        scanf ( "%f", &b[i] ) ;
        printf ( "\nangle%d = ", i+1 ) ;
        scanf ( "%f", &angle[i] ) ;

        /* Calculate area */
        area[i] = ( 1.0 / 2 ) * a[i] * b[i] * sin ( angle[i] ) ;

        /* Note the largest value of area */
        if ( area[i] > largest )
        {
            largest = area[i] ;
            plot = i ; /* note the element with largest value of area */
        }
        clrscr( ) ;
    }
    clrscr( ) ;

    /* Print area of all plots */
    printf ( "\n\nEntered Plot dimensions and respective Area is:\n" ) ;
    printf ( "\nPlot No.\ta\tb\tAngle\tArea" ) ;
    for ( i = 0 ; i <= 5 ; i++ )
    {
        printf ( "\n%d\t\t%.2f\t%.2f\t%.2f\t%.3f ", i+1, a[i], b[i], angle[i],
```

```
                area[i] ) ;
    }
    printf ( "\n\nLargest Triangular Area = %.3f having values of a, b and
            angle as:", largest ) ;
    printf ( "\n\na = %.2f\tb = %.2f\tangle = %.2f ", a[plot], b[plot],
            angle[plot] ) ;

    getch( ) ;
}
```

[53] For the following set of n data points (x, y), compute the correlation coefficient r, given by

$$r = \frac{\sum xy - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$$

x	y
34.22	102.43
39.87	100.93
41.85	97.43
43.23	97.81
40.06	98.32
53.29	98.32
53.29	100.07
54.14	97.08
49.12	91.59
40.71	94.85
55.15	94.65

Program:

```
/* Calculation corelation coefficient */

#include <stdio.h>
#include <conio.h>
```



```
#include <math.h>

main( )
{
    int n = 11, i, j ;
    float x[11] = { 34.22, 39.87, 41.85, 43.23, 40.06, 53.29, 53.29,
                    54.14, 49.12, 40.71, 55.15 } ;
    float y[11] = { 102.43, 100.93, 97.43, 97.81, 98.32, 98.32, 100.07,
                    97.08, 91.59, 94.85, 94.65 } ;
    float r ;
    float sx = 0.0, sxy = 0.0, sy = 0.0, sxs = 0.0, sys = 0.0 ;
    float r1, r2 ;

    clrscr( ) ;

    for ( i = 0 ; i <= 10 ; i++ )
    {
        sx = sx + x[i] ; /* Summation of X */
        sxy = sxy + x[i] * y[i] ; /* Summation of XY */
        sy = sy + y[i] ; /* Summation of Y */
        sxs = sxs + x[i] * x[i] ; /* Summation of square of X */
        sys = sys + y[i] * y[i] ; /* Summation of square of Y */
    }
    printf ( "\n\n%c X   = %.2f ", 228, sx ) ;
    printf ( "\n%c Y   = %.2f ", 228, sy ) ;
    printf ( "\n%c XY  = %.2f ", 228, sxy ) ;
    printf ( "\n%c X*X = %.2f ", 228, sxs ) ;
    printf ( "\n%c Y*Y = %.2f ", 228, sys ) ;
    r1 = ( sxy - sx * sy ) ;
    r2 = ( sqrt ( ( n * sxs - sx * sx ) * ( n * sys - sy * sy ) ) ) ;
    r = r1 / r2 ;
    printf ( "\n\n( Numerator ) r1  = %f", r1 ) ;
    printf ( "\n( Denominator) r2 = %f", r2 ) ;
    printf ( "\n\n Corelation coefficient r1/r2 = %f ", r ) ;
    getch( ) ;
}
```

[54] For the following set of point given by **(x, y)** fit a straight line given by

y = a + bx

where,

$a = \bar{y} - b\bar{x}$ *and*

$$b = \frac{n\sum yx - \sum x \sum y}{[n\sum x^2 - (\sum x)^2]}$$

x	y
3.0	1.5
4.5	2.0
5.5	3.5
6.5	5.0
7.5	6.0
8.5	7.5
8.0	9.0
9.0	10.5
9.5	12.0
10.0	14.0

Program:

```
/* Equation of a straight line */

#include <stdio.h>
#include <conio.h>
#include <math.h>
main( )
{
    int n = 10, i ;
    float x[10] = { 3.0, 4.5, 5.5, 6.5, 7.5, 8.5, 8.0, 9.0, 9.5, 10.0 } ;
    float y[10] = { 1.5, 2.0, 3.5, 5.0, 6.0, 7.5, 9.0, 10.5, 12.0, 14.0 } ;
```



```
    float a, b ;
    float sx = 0.0, sxy = 0.0, sy = 0.0, sxs = 0.0 ;
    float r1, r2, mx = 0.0, my = 0.0, yy ;

    clrscr( ) ;
    for ( i = 0 ; i <= 9 ; i++ )
    {
        sx = sx + x[i] ;  /* Summation of X */
        sxy = sxy + x[i] * y[i] ;  /* Summation of XY */
        sy = sy + y[i] ;  /* Summation of Y */
        sxs = sxs + x[i] * x[i] ;  /* Summation of square of X */
    }

    printf ( "\n\n%c X  = %.2f " , 228, sx ) ;
    printf ( "\n%c Y  = %.2f " , 228, sy ) ;
    printf ( "\n%c XY  = %.2f " , 228, sxy ) ;
    printf ( "\n%c X*X = %.2f " ,228, sxs ) ;

    r1 = ( n * sxy - sx * sy ) ;
    r2 = ( ( n * sxs ) - ( sx * sx ) ) ;
    b = r1 / r2 ;

    printf ( "\n\nr1 = %f", r1 ) ;
    printf ( "\nr2 = %f", r2 ) ;
    printf ( "\n\nValue of b  = %f ", b ) ;

    mx = sx / n ;  /* mean of x */
    my = sy / n ;  /* mean of y */
    a = my - b * mx ;
    printf ( "\nValue of a = %f ", a ) ;
    printf ( "\\nnEquation of the line is : Y = %.2f* X %+.2f ", b , a ) ;

    getch( ) ;
}
```

[55] Fit an exponential curve of the form $y = ax^b$ for the above data points.

Program:

```
/* Equation of an exponential curve */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    int  n = 10, i ;
    float  x[10] = { 3.0, 4.5, 5.5, 6.5, 7.5, 8.5, 8.0, 9.0, 9.5, 10.0 } ;
    float  y[10] = { 1.5, 2.0, 3.5, 5.0, 6.0, 7.5, 9.0, 10.5, 12.0, 14.0 } ;
    float  a, b ;
    float  sx = 0.0, sxy = 0.0, sy = 0.0, sxs = 0.0 ;
    float  r1, r2, mx = 0.0, my = 0.0, yy ;

    clrscr( ) ;

    for ( i = 0 ; i <= 9 ; i++ )
    {
        sx = sx + x[i] ;
        sxy = sxy + x[i] * y[i] ;
        sy = sy + y[i] ;
        sxs = sxs + x[i] * x[i] ;
    }

    printf ( "\n\n%c X    = %.2f " ,228, sx ) ;
    printf ( "\n%c Y    = %.2f " , 228, sy ) ;
    printf ( "\n%c XY   = %.2f " , 228, sxy ) ;
    printf ( "\n%c (X*X) = %.2f\n " , 228, sxs ) ;

    r1 = ( n * sxy - sx * sy ) ;
    r2 = ( ( n * sxs ) - ( sx * sx ) ) ;
    b = r1 / r2 ;

    printf ( "\nr1 = %.2f", r1 ) ;
```



```
    printf ( "\nr2 = %.2f", r2 ) ;

    mx = sx / n ; /* mean of x */
    my = sy / n ; /* mean of y */
    a = my - b * mx ;

    printf ( "\n\nValue of a = %f ", a ) ;
    printf ( "\n\nValue of b  = %f ", b ) ;

    printf ( "\n\nEquation of Exponential Curve is : " ) ;
    printf ( "Y = %+.2f * ( X ** %+.2f ) ", a, b ) ;
    getch( ) ;
}
```

[56] The **X** and **Y** coordinates of 10 different points are entered through the keyboard. Write a program to find the distance of last point from the first point. (Sum of distance between consecutive Point).

Program:

```
/* Distance between two points */

#include <stdio.h>
#include <conio.h>
#include <math.h>

main( )
{
    int  i, x[10], y[10] ;
    float  sum = 0 ;

    clrscr( ) ;

    printf ( "\nEnter the coordinates for the points:" ) ;
    for ( i = 0 ; i <= 9 ; i++ )
```

```
    {
        printf ( "\nX[%d] = ", i+1 ) ;
        scanf ( "%d", &x[i] ) ;

        printf ( "Y[%d] = ", i+1 ) ;
        scanf ( "%d", &y[i] ) ;
    }

    for ( i = 0 ; i <= 8 ; i++ )
    {
        sum = sum + sqrt ( pow ( ( x[i+1] - x[i] ), 2 ) + pow ( ( y[i+1] -
                y[i] ), 2 ) ) ;
    }

    printf ( "\n\nDistance of the last point from the first point = %.2f ",
            sum ) ;
    getch( ) ;
}
```

Structures

[57] Create a structure to specify data on students given below:

Roll number, Name, Department, Course, Year of joining
Assume that there are not more than 450 students in the collage.

(a) Write a function to print names of all students who joined in a particular year.

(b) Write a function to print the data on a student whose roll number is given.

Program:

```
/* create structure to hold student data */

#include <stdio.h>
```



```
#include <conio.h>

struct stud
{
    int r_n ; /* Roll Number */
    char name[20] ; /* Name */
    char dep[15] ; /* Department */
    char course[10] ; /* Course */
    int y_o_j ; /* Year Of Joining */
};
struct stud s[450] ; /* array of structure */

main( )
{
    int i, r ;
    int y ;

    clrscr( ) ;

    printf ( "\nEnter the data for each Student:\n\n " ) ;

    /* Initialize the values for the students structure */
    set_student_data( ) ;

    /* Display all the elements of the student structure */
    display( ) ;

    /* Search data on year of Joining */
    printf ( "\nEnter the Year of Joining of the Student " ) ;
    scanf ( "%d", &y ) ;

    name_acc_year ( y ) ; /* year of joining passed to function */

    /* Search data based on roll number */
    printf ( "\nEnter the Roll Number of the Student" ) ;
    scanf ( "%d", &r ) ;

    data_acc_rollno ( r ) ; /* roll number passed to function */
```

```
}

set_student_data( )  /* Enter student data */
{
    int i ;
    for ( i = 0 ; i < 450 ; i++ )
    {
        fflush ( stdin ) ;  /* Flush the input buffer */
        printf ( "\nEnter the Roll NUmber of the student\n" ) ;
        scanf ( "%d", &s[i].r_n ) ;
        fflush ( stdin ) ;
        printf ( "Enter the name of the student\n" ) ;
        scanf ( "%s", s[i].name ) ;
        fflush ( stdin ) ;
        printf ( "Enter the name of the Department\n" ) ;
        scanf ( "%s", s[i].dep ) ;
        fflush ( stdin ) ;
        printf ( "Enter the name of the Course\n" ) ;
        scanf ( "%s", s[i].course ) ;
        fflush ( stdin ) ;
        printf ( "Enter the Year of Joining of the student\n" ) ;
        scanf ( "%d", &s[i].y_o_j ) ;
        clrscr( ) ;
    }
}

/* function to display data */
display( )
{
        int i ;
        for ( i = 0 ; i < 450 ; i++ )
        {
            printf ( "\n\tRoll Number of student %d =  %d \n", i+1 ,
                    s[i].r_n ) ;
            printf ( "\n\tName of student %d = %s \n",i+1, s[i].name ) ;
            printf ( "\n\tName of the Department = %s \n", s[i].dep ) ;
            printf ( "\n\tName of the Course = %s \n", s[i].course ) ;
            printf ( "\n\tYear of Joining of student %d = %d \n\n", i+1 ,
```



```
                    s[i].y_o_j ) ;
            getch( ) ;
        }
}

/* function to get name based on year of joining */
name_acc_year ( int  y )
{
    int  i, j = 0 ;

    for ( i = 0 ; i < 450 ; i++ )
    {
        if ( y == s[i].y_o_j )
        {
            printf ( "%s joined in the year %d\n", s[i].name, s[i].y_o_j ) ;
            j = 1 ;
        }
    }
    if ( j == 0 )
        printf ( "\nNo student joined in the year %d", y ) ;
    getch( ) ;
}

/* function to get student data based on roll number */
int  data_acc_rollno ( int  r )
{
    int  i, j = 0 ;
    for ( i = 0 ; i < 450 ; i++ )
    {
        if ( s[i].r_n == r )
        {
            printf ( "\n\tRoll Number of student =  %d \n", s[i].r_n ) ;
            printf ( "\n\tName of student = %s \n", s[i].name ) ;
            printf ( "\n\tName of the Department = %s \n", s[i].dep ) ;
            printf ( "\n\tName of the Course = %s \n", s[i].course ) ;
            printf ( "\n\tYear of Joining of student = %d \n\n", s[i].y_o_j )
```

```
            j = 1 ;
        }
    }
    if ( j == 0 )
        printf ( "\nNo such Roll Number present." ) ;
    getch( ) ;
}
```

[58] Create a structure to specify data of customers in a bank. The data to be stored is: Account number, Name, Balance in account. Assume maximum of 200 customers in the bank.

(a) Write a function to print the Account number and name of each customer with balance below Rs. 100
If a customer request for withdrawal or deposit, it is given in the form:
Acct. no, amount, (1 for deposit, 0 for withdrawal)

(b) Write a program to give a message, "the balance is insufficient for the specified withdrawal".

Program:

```
/* Creat structure of 200 customer's banks data */

#include <stdio.h>
#include <conio.h>

/* function to link the floating point format*/
fun( )
{
    float f, *ff = &f ;
}

/* structure for customer */
struct customer
{
    int acc_no ;
    char name[20] ;
    float bal_i_acc ;
} ;
struct customer cust[200] ;

/* function prototypes */
int withdrawal ( int ano, float amount ) ;
int with_dep ( int ano, float amount ) ;
int deposit ( int ano, float amount ) ;

main( )
{
    int i, ano, choice ;
    float amount ;

    clrscr( ) ;
    /*Function used to initialise the Customer record*/
    set_cust_data( ) ;
    /*Function used to display the record of all the customers*/
    display( ) ;

    /* Function to print name and account number of the customer
       Whose balance is less then 100*/
    low_bal( ) ;
    /* Function to withdraw or deposit the amount in the account
       Number of the specified customer*/

    printf ( "\nEnter the account number and the amount to be
              deposited/withdrawn" ) ;
    scanf ( "%d%f", &ano, &amount ) ;
    with_dep ( ano, amount ) ;

    /*display all the records of the customers*/
    display( ) ;
}
with_dep ( int ano, float amount )
{
    int choice ;
    printf ( "Enter 1 for Deposit\t \tEnter 0 for Withdrawal" ) ;
```

```
    scanf ( "%d", &choice ) ;
    switch ( choice )
    {
        case 1 :
            deposit ( ano, amount ) ;
            break ;
        case 0 :
            withdrawal ( ano, amount ) ;
            break ;
        default :
            printf ( "\nYou entered wrong choice" ) ;
    }
}
deposit ( int  ano, float  amount )
{
    int  i, j = 0 ;
    for ( i = 0 ; i < 200 ; i++ )
    {
        if ( cust[i].acc_no == ano )
        {
            cust[i].bal_i_acc += amount ;
            j = 1 ;
        }
    }
    if ( j == 0 )
        printf ( "\n\tWrong Account Number" ) ;
}
withdrawal ( int  ano, float  amount )
{
    int  i, j = 0 ;
    for ( i = 0 ; i < 200 ; i++ )
    {
        if ( cust[i].acc_no == ano )
        {
            j = 1 ;
            if ( cust[i].bal_i_acc < 100 )
            {
                printf ( "\n\t\tThe Balance is insufficient for the
```



```
                                specified Withdrawal" ) ;
                getch( ) ;
            }
            else
                if ( cust[i].bal_i_acc - 100 >= amount )
                    cust[i].bal_i_acc -= amount ;
                else
                    printf ( "\nwithdrawal amount should be less than
                            or equal to %f Rs.", cust[i].bal_i_acc - 100 ) ;
        }
    }
    if ( j == 0 )
        printf ( "\n\tWrong Account Number" ) ;
}

low_bal( )
{
    int  i, j = 0 ;
    clrscr( ) ;
    printf ( "\n\nName and Account number of the customers \nwhose
            balance is less than 100" ) ;
    for ( i = 0 ; i < 200 ; i++ )
    {
        if ( cust[i].bal_i_acc < 100 )
        {
            j = 1 ;
            printf ( "\n\tCustomer Number = %d", i+1 ) ;
            printf ( "\n\t\tAccount Number of Customer = %d",
                    cust[i].acc_no ) ;
            printf ( "\n\t\tName of Customer = %s", cust[i].name ) ;
        }
    }
    if ( j == 0 )
        printf ( "\n\tEvery Customer has more than minimum balance
                amount" ) ;
    getch( ) ;
}
display( )
```


www.indianfun.org

```
    {
        int i ;
        for ( i = 0 ; i < 200 ; i++ )
        {
            printf ( "\n\n\tCustomer Number = %d", i+1 ) ;
            printf ( "\n\n\tAccount Number of Customer = %d",
                    cust[i].acc_no ) ;
            printf ( "\n\tName of Customer = %s", cust[i].name ) ;
            printf ( "\n\tBalance Amount of Customer = %.3f",
                    cust[i].bal_i_acc ) ;
        }
        getch( ) ;
    }
    set_cust_data( )
    {
        int i ;
        for ( i = 0 ; i < 200 ; i++ )
        {
            printf ( "\n\nEnter the Account number of the Customer\n\t" ) ;
            scanf ( "%d", &cust[i].acc_no ) ;

            fflush ( stdin ) ;
            printf ( "\nEnter the name of the Customer\n\t" ) ;
            scanf ( "%s", cust[i].name ) ;

            fflush ( stdin ) ;
            printf ( "\nEnter the balance amount in the Customer
                    account\n\t" ) ;
            scanf ( "%f", &cust[i].bal_i_acc ) ;
            clrscr( ) ;
        }
    }
```

[59] An automobile company has serial number for engine parts starting from AA0 to FF9. The other characteristics of parts to be specified in a structure are: Year of manufacture, material and quantity manufactured.

(a) Specify a structure to store information corresponding t a part.

(b) Write a program to retrieve information on parts with serial numbers between BB1 and CC6.

Program:

```
/* Create structure to store engine parts data */

#include <conio.h>
#include <stdio.h>

struct automobile
{
    int s_no ;
    int year_o_manu ;
    char material[20] ;
    int quantity ;
} ;
struct automobile part[2] ;

main( )
{
    int i ;

    clrscr( ) ;
    /* Function to set the values for the records*/
    set_auto_data( ) ;

    clrscr( ) ;
    /*List all the records*/
    display( ) ;

    clrscr( ) ;
    /* Function to retrieve information on parts with serial number
       between BB1 and CC6*/
```

```
    retrieve( ) ;

}

retrieve( )
{
    int i, j = 0 ;
    printf ( "\nList of parts between BB1 & CC6:" ) ;
    for ( i = 0 ; i < 2 ; i++ )
    {
        if ( ( part[i].s_no >= 0xbb1 ) && ( part[i].s_no <= 0xcc6 ) )
        {
            j = 1 ;
            printf ( "\n\n\tPart Number = %d", i ) ;
            printf ( "\nSerial Number = %x", part[i].s_no ) ;
            printf ( "\nYear of manufacturing = %d",
                        part[i].year_o_manu ) ;
            printf ( "\nMaterial used : %s", part[i].material ) ;
            printf ( "\nManufacture Quantity = %d", part[i].quantity ) ;
        }
    }
    if ( j == 0 )
        printf ( "\nNo such record prsent" ) ;
    getch( ) ;
}
display( )
{
    int i ;
    for ( i = 0 ; i < 2 ; i++ )
    {
        printf ( "\n\n\tPart Number = %d", i ) ;
        printf ( "\nSerial Number = %x", part[i].s_no ) ;
        printf ( "\nYear of manufacturing = %d", part[i].year_o_manu ) ;
        printf ( "\nMaterial used : %s", part[i].material ) ;
        printf ( "\nManufacture Quantity = %d", part[i].quantity ) ;
    }
    getch( ) ;
}
```



```
set_auto_data( )
{
    int i ;
    for ( i = 0 ; i < 2 ; i++ )
    {
        while ( 1 )
        {
            printf ( "\nEntér the serial Number of the part" ) ;
            printf ( "\nNumber must be between AA0 and FF9 " ) ;
            scanf ( "%x", &part[i].s_no ) ;
            if ( part[i].s_no >= 0xAA0 && part[i].s_no <= 0xff9 )
                break ;
            clrscr( ) ;
        }
        printf ( "\nEnter the Year of manufacturing of the part" ) ;
        scanf ( "%d", &part[i].year_o_manu ) ;

        printf ( "\nEnter the material of the part" ) ;
        scanf ( "%s", &part[i].material ) ;
        fflush ( stdin ) ;

        printf ( "\nEnter the quantity of the part" ) ;
        scanf ( "%d", &part[i].quantity ) ;
    }
}
```

[60] Linked list is a very common data structure often used to store similar data in memory. While the element of an array occupy contiguous memory locations, those of a linked list are not constrained to be stored in adjacent location. The individual element are stored "somewhere" in memory, rather like a family dispersed, but still bound together. The order of the elements is a collection of element called nodes, each of which stores two item of information: one, an element of the list, and two, a link, i.e. a pointer or an address that indicates explicitly the location of the node containing the successor of this list element.

www.indianfun.org

Write a program to build a linked list by adding new nodes at the beginning, at the end or in the middle of the linked list. Also write a function **display( )** which display all the nodes present in the linked list.

Program:

```
/* Build a Linked List */

#include <stdio.h>
#include <malloc.h>

/* structure definition */
struct linklist
{
    int  item ;
    struct linklist *link ;
} ;

struct linklist *p ;

main( )
{
    p = NULL ; /* Last node link is always NULL */
    clrscr( ) ;

    /*Function to add the item to the beginning of the linklist*/
    add ( 10 ) ;
    add ( 20 ) ;
    add ( 30 ) ;
    add ( 40 ) ;
    printf ( "Items added at the begining of the list:\n" ) ;
    /*Function to display all the items in the linklist*/
    display( ) ;
    printf ( "  Press any key ... " ) ;
    getch( ) ;
```



```
    /*Function to add item at the end of the linklist*/
    printf ( "\n\nItem added at the end of list:\n" ) ;
    add_end ( 50 ) ;
    display( ) ;
    printf ( "  Press any key ... " ) ;
    getch ( ) ;

    /* Function to add item at a given location */
    printf ( "\n\nItem added in the middle of the list:\n" ) ;
    add_item_at_loca ( 60, 4 ) ;
    display( ) ;
    printf ( "  Press any key ... " ) ;
    getch( ) ;
}
add_item_at_loca ( int  item, int  loca )
{
    struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
    struct linklist *r, *w = p ;
    int  i = 1 ;
    q -> item = item ;
    q -> link = NULL ;
    if ( p == NULL )
        p = q ;
    else
    {
        while ( ( w->link != NULL ) && ( i != loca ) )
        {
            i++ ;
            w = w -> link ;
        }
        if ( i == loca )
        {
            r = w -> link ;
            w -> link = q ;
            q -> link = r ;
        }
        else
            w -> link = q ;
```


www.indianfun.org

```
    }
}
add_end ( int  item )
{
    struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
    struct linklist *w = p ;
    q -> item = item ;
    q -> link = NULL ;
    if ( p == NULL )
        p = q ;
    else
    {
        while ( w -> link != NULL )
            w = w -> link ;
        w -> link = q ;
    }
}
display( )
{
    struct linklist *q = p ;
    int  i = 0 ;
    while ( q != NULL )
    {
        i++ ;
        printf ( "\nitem number %d = %d ", i , q->item ) ;
        q = q -> link ;
    }
}
add( int  item )
{
    struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
    q -> item = item ;
    q -> link = NULL ;
    if ( p == NULL )
        p = q ;
    else
    {
        q -> link = p ;
```



```
        p = q ;
    }
}
```

[61] Add a function **delete( )** to program[5] above, which can delete any node in the linked list.

Program:

```
/* Delete from a Linked List */

#include <stdio.h>
#include <malloc.h>

struct linklist
{
    int  item ;
    struct linklist *link ;
} ;
struct linklist *p ;

main( )
{
    p = NULL ;
    clrscr( ) ;

    /*Function to add the item to the beginning of the linklist*/
    add ( 10 ) ;
    add ( 20 ) ;
    add ( 30 ) ;
    add ( 40 ) ;
    printf ( "Items added at the begining of the list:\n" ) ;
    /*Function to display all the items in the linklist*/
    display( ) ;
    printf ( "   Press any key ..." ) ;
    getch( ) ;
```

```
    /*Function to delete the item from the linklist*/
    deleteitem ( 40 ) ;
    printf ( "\n\nItem deleted from the list:\n" ) ;
    display( ) ;
    printf ( "   Press any key ..." ) ;
    getch( ) ;

    /*Function to add item at the end of the linklist*/
    printf ( "\n" ) ;
    add_end ( 50 ) ;
    printf ( "\n\nItem added at the end of list:\n" ) ;
    display( ) ;
    printf ( "   Press any key ..." ) ;
    getch( ) ;

    printf ( "\n" ) ;
    add_item_at_loca ( 60, 3 ) ;
    printf ( "\n\nItem added in the middle of the list:\n" ) ;
    display( ) ;
    printf ( "   Press any key ..." ) ;
    getch( ) ;
}
deleteitem ( int  item )
{
    struct linklist *w, *q = p ;
    int  i = 0 ;
    if ( p == NULL )
        printf ( "linklist is empty" ) ;
    else
    {
        while ( q != NULL )
        {
            i++ ;
            if ( q -> item == item )
            {
                if ( i == 1 )
                {
                    w = q ;
```



```
                    q = q -> link ;
                    p = q ;
                    free ( w ) ;
                    break ;
                }
                w = p ;
                while ( w -> link != q )
                    w = w -> link ;
                w -> link = q -> link ;
                free( q ) ;
                break ;
            }
            q = q -> link ;
        }
    }
}
add_item_at_loca ( int  item, int  loca )
{
    struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
    struct linklist *r, *w = p ;
    int  i = 1 ;
    q -> item = item ;
    q -> link = NULL ;
    if ( p == NULL )
        p = q ;
    else
    {
        while ( ( w->link != NULL ) && ( i != loca ) )
        {
            i++ ;
            w = w -> link ;
        }
        if ( i == loca )
        {
            r = w -> link ;
            w -> link = q ;
            q -> link = r ;
        }
```


www.indianfun.org

```
            else
                w -> link = q ;
        }
    }
    add_end ( int  item )
    {
        struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
        struct linklist *w = p ;
        q -> item = item ;
        q -> link = NULL ;
        if ( p == NULL )
            p = q ;
        else
        {
            while ( w -> link != NULL )
                w = w -> link ;
            w -> link = q ;
        }
    }
    display( )
    {
        struct linklist *q = p ;
        int  i = 0 ;
        while ( q != NULL )
        {
            i++ ;
            printf ( "\nitem number %d = %d ", i , q->item ) ;
            q = q -> link ;
        }
    }
    add ( int  item )
    {
        struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
        q -> item = item ;
        q -> link = NULL ;
        if ( p == NULL )
            p = q ;
        else
```



```
        {
            q -> link = p ;
            p = q ;
        }
    }
```

[62] Write a program to maintain a linked list such that that every element added to the linked list gets inserted at such a place that the linked list is always maintained in ascending order.

Program:

```
/* Insertion in a sorted linked list */

#include <stdio.h>
#include <conio.h>
#include <malloc.h>

struct linklist
{
    int  item ;
    struct linklist *link ;
};
struct linklist *p ;

main( )
{
    int  n ;
    p = NULL ;
    clrscr( ) ;

    /* Add the item to the beginning of the linked list */
    add ( 40 ) ;
    add ( 5 ) ;
    add ( 3 ) ;
    add ( 10 ) ;
```

```
    /* Display all the items in the linked list */
    display( ) ;

    /* Insert an item entered through keyboard  at a proper place */
    printf ( "\n\nEnter an integer to be added to Linked List: " ) ;
    scanf ( "%d" , &n ) ;
    add ( n ) ;
    display( ) ;

    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* Function to display all the items in the linked list */
display( )
{
    struct linklist *q = p ;
    int  i = 0 ;
    while ( q != NULL )
    {
        i++ ;
        printf ( "\nitem number %d = %d ", i , q -> item ) ;
        q = q -> link ;
    }
}
/* Function to add the item to the beginning of the linked list */
add ( int  item )
{
    struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
    struct linklist *p1 ;
    int  i = 0 ;

    q -> item = item ;  /* new node */
    q -> link = NULL ;  /* link of new node set to NULL */
    if ( p == NULL )
        p = q ;  /* create first node */
    else
    {
```



```
        p1 = p ;  /* loop counter */
        while ( p1 != NULL )
        {
            i++ ;
            if ( p1 -> item < item )  /* compare new item with existing */
            {
                if ( p1 -> link == NULL )
                {
                    p1 -> link = q ;  /* add second node at the end */
                    break ;
                }
                if ( p1 -> link -> item >= item )
                {
                    q -> link = p1 -> link ;  /* add node in the middle */
                    p1 -> link = q ;
                    break ;
                }
            }
            else
            {
                if ( i == 1 )
                {
                    q -> link = p ;  /* add second node at the
                                        beginning */
                    p = q ;
                }
            }
            p1 = p1 -> link ;
        }
    }
}
```

[63] Write a program to reverse the links in the existing linked list such that the last node becomes the first node and the first becomes the last.

Program:

```
/* Reversing the links in the linked list */

#include <stdio.h>
#include <conio.h>
#include <malloc.h>

struct linklist
{
    int item ;
    struct linklist *link ;
};
struct linklist *p ;

main( )
{
    p = NULL ;
    clrscr( ) ;

    /* Add the item to the beginning of the linked list */
    add ( 1 ) ;
    add ( 2 ) ;
    add ( 3 ) ;
    add ( 4 ) ;
    add ( 10 ) ;
    add ( 20 ) ;
    add ( 30 ) ;
    add ( 40 ) ;

    /* Display all the items in the linked list */
    printf ( "\nLinked list as entered:" ) ;
    display( ) ;
    getch( ) ;

    /* Reverse the linked list */
    reverse( ) ;
    printf ( "\n\nLinked list after reversing:" ) ;
    display( ) ;
    printf ( "\n\n\nPress any key to exit..." ) ;
```



```
    getch( ) ;
}

/* Function to reverse the linked list */
reverse( )
{
    struct linklist *q = p, *r, *w = NULL ;
    int i = 1 ;
    if ( p != NULL )
    {
        while ( q != NULL )
        {
            r = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
            r -> item = q -> item ;
            if ( i == 1 )
                r -> link = w ;
            else
                r -> link = w ;
            w = r ;
            i++ ;
            q = q -> link ;
            free ( p ) ;
            p = q ;
        }
        p = w ;
    }
    else
        printf ( "\nLinked list is empty" ) ;
}

/* Function to display all the items in the linked list */
display( )
{
    struct linklist *q = p ;
    int i = 0 ;
    printf ( "\n" ) ;
    while ( q != NULL )
    {
```

```
        i++ ;
        printf ( "\nitem number %d = %d ", i , q -> item ) ;
        q = q -> link ;
    }
}

/* Function to add the item to the beginning of the linked list */
add ( int  item )
{
    struct linklist *q = ( struct linklist * ) malloc ( sizeof ( struct linklist ) ) ;
    struct linklist *p1 ;
    int  i = 0 ;

    q -> item = item ;
    q -> link = NULL ;
    if ( p == NULL )
        p = q ;
    else
    {
        q -> link = p ;
        p = q ;
    }
}
```

[64] A stack is a data structure in which addition of few element or deletion of existing element always takes place at the same end. This end is often known as 'top' of stack. This situation can be compared to a stack of plates in a cafeteria where every new plate taken off the stack is also from the 'top' of the stack. There are several application where stack can be put to use. For example, recursion, keeping track of function calls, evaluation of expressions etc. Write a program to implement a stack using a linked list.

Program:


```
/* Implementation of stack using  a linked list */

#include <alloc.h>
struct node
{
    int  data ;
    struct node *link ;
} ;
void displaystack ( struct node *q ) ;

main( )
{
    struct node *top ;
    int  t, i, item ;

    clrscr( ) ;
    top = NULL ;

    /* Insert items in the stack */
    push ( &top, 45 ) ;
    push ( &top, 28 ) ;
    push ( &top, 63 ) ;
    push ( &top, 55 ) ;

    /* Display items on the stack */
    displaystack ( top ) ;
    t = count ( top ) ;
    printf ( "\nTotal items = %d\n" , t ) ;

    printf ( "\nPress any key...\n" ) ;
    getch( ) ;

    /* Remove an item from stack */
    printf ( "\nPopped : " ) ;
    item = pop ( &top ) ;
    printf ( "%d\n", item ) ;

    displaystack ( top ) ;
```

```
    t = count ( top ) ;
    printf ( "\nTotal items = %d" , t ) ;
    getch( ) ;
}

/* Function to add items to stack */
int  push ( struct node **s, int  item )
{
    struct node *q ;
    q = ( struct node * ) malloc ( sizeof ( struct node ) ) ;
    q -> data = item ;
    q -> link = *s ;
    *s = q ;
}

/* Function to remove items from stack */
int  pop ( struct node **s )
{
    int  item ;
    struct node *q ;
    if ( *s == NULL )
        printf ( "Stack is empty" ) ;
    else
    {
        q = *s ;
        item = q -> data ;
        *s = q -> link ;
        free ( q ) ;
        return ( item ) ;
    }
}

/* Function to display contents of stack */
void displaystack ( struct node *q )
{
    while ( q != NULL )
    {
        printf ( "\t%2d ", q -> data ) ;
```



```
        q = q -> link ;
    }
}

/* Function to keep count of items */
int  count ( struct node * q )
{
    int  c = 0 ;
    while ( q != NULL )
    {
        q = q -> link ;
        c++ ;
    }
    return c ;
}
```

[65] Unlike a stack, in a queue the addition of new element takes place at the end (called 'rear' of queue) whereas deletion takes place at the other end (called "front" of queue). Write a program to implement a queue using a linked list.

Program:

```
/* Implementation of a queue using linked list */

#include <stdio.h>
#include <conio.h>
#include <malloc.h>

struct queue
{
    int  item ;
    struct queue *link ;
} ;
struct queue *rear, *front ;

main( )
```


www.indianfun.org

```
{
    int item ;
    rear = front = NULL ;
    clrscr( ) ;

    /* Add an item to the queue */
    add ( 10 ) ;
    add ( 20 ) ;
    add ( 30 ) ;
    add ( 40 ) ;
    add ( 50 ) ;
    add ( 60 ) ;

    /* Display all the elements of the queue */
    display( ) ;
    printf ( "\nTotal number of elements present is : %d ", count( ) ) ;
    getch( ) ;

    /* Delete an element from a Queue */
    item = del_queue( ) ;
    printf ( "\nDeleted Item = %d\n", item ) ;
    display( ) ;
    printf ( "\nTotal number of elements present is : %d ", count( ) ) ;
    getch( ) ;

    item = del_queue( ) ;
    printf ( "\nDeleted Item = %d\n", item ) ;
    display( ) ;
    printf ( "\nTotal number of elements present is : %d ", count( ) ) ;
    getch( ) ;

    item = del_queue( ) ;
    printf ( "\nDeleted Item = %d\n", item ) ;
    display( ) ;
    printf ( "\nTotal number of elements present is : %d ", count( ) ) ;
    getch( ) ;
}
```



```
/* Function to add an item to the queue */
add ( int item )
{
    struct queue *q = ( struct queue * ) malloc ( sizeof ( struct queue ) ) ;
    q -> item = item ;
    q -> link = NULL ;
    if ( rear == NULL )
    {
        rear = q ;
        front = q ;
    }
    else
    {
        q -> link = rear ;
        rear = q ;
    }
}

/* Function to delete an element from a Queue */
del_queue( )
{
    int item ;
    struct queue *q = rear ;
    if ( front == NULL )
    {
        printf ( "\n\n\t\tQueue is empty" ) ;
        getch( ) ;
        return -1;
    }
    else
    {
        if ( front == rear )
        {
            item = q -> item ;
            front = rear = NULL ;
            free( q ) ;
        }
        else
```

```
        {
            while( q -> link -> link != NULL )
                q = q -> link ;
            item = q -> link -> item ;
            free( q -> link ) ;
            front = q ;
            q -> link = NULL ;
        }
    }
    return item ;
}

/* Function to display all the elements of the queue */
display( )
{
    struct queue *q = rear ;
    while ( q != NULL )
    {
        printf ( "\n%d", q -> item ) ;
        q = q -> link ;
    }
}
count( )
{
    struct queue *q = rear ;
    int count = 0 ;
    while ( q != NULL )
    {
        count ++ ;
        q = q -> link ;
    }
    return count ;
}
```

[66] A record contains name of cricketer, his age, number of test matches that he has played and the average runs that he has scored in each test match. Create an array of structure to hold records of 20 such cricketer and then write a program to read

these records and arrange them in ascending order by average runs. Use the **qusort( )** standard library function.

Program.

```
/* Create array of structures, sort and display */

#include <malloc.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

/* Function to link the floating point format */
fun( )
{
    float f, *ff = &f ;
    return f ;
}

struct cric_player
{
    char name[20] ;
    int age ;
    int notest ;
    float avgrun ;
} ;
struct cric_player cp[3] ;
int sort_function ( const void *, const void * ) ;

main( )
{
    /* Set the values of the structure elements */
    setdata( ) ;

    /* Display all the entries present in the structure */
    clrscr( ) ;
    printf ( "\nData as entered : \n" ) ;
```

```
        display( ) ;
        getch( ) ;

        /* Sorting the array of structures */
        sortbyavg( ) ;
        clrscr( ) ;
        printf ( "\nData sorted on Average Runs :\n" ) ;
        display( ) ;
        getch( ) ;
    }

    /* Function used for sorting the array of structure */
    sortbyavg( )
    {
        int i, j ;
        struct cric_player t ;
        qsort ( ( struct cric_player * ) cp, 3, sizeof ( cp[0] ), sort_function ) ;
    }

    int sort_function ( const void *f, const void *ff )
    {
        return ( ( ( struct cric_player *) f ) -> avgrun - ( ( struct cric_player * )
                ff ) -> avgrun ) ;
    }

    /* Function to display all the entries present in the sturcture */
    display( )
    {
        int i ;
        for ( i = 0 ; i < 3 ; i++ )
        {
            printf ( "\n\n\tName : %s", cp[i].name ) ;
            printf ( "\n\tAge : %d", cp[i].age ) ;
            printf ( "\n\tNo of tests : %d", cp[i].notest ) ;
            printf ( "\n\tAverage : %f", cp[i].avgrun ) ;
        }
    }
```



```
    /* Function to set the values of the structure */
    setdata( )
    {
        int i ;
        for ( i = 0 ; i < 3 ; i++ )
        {
            clrscr( ) ;
            printf ( "\nEnter the name:" ) ;
            scanf ( "%s", &cp[i].name ) ;
            fflush ( stdin ) ;

            printf ( "\nEnter the age:" ) ;
            scanf ( "%d", &cp[i].age ) ;
            fflush ( stdin ) ;

            printf ( "\nEnter the total number of test matches played:" ) ;
            scanf ( "%d", &cp[i].notest ) ;
            fflush ( stdin ) ;

            printf ( "\nEnter the avgerage number of runs scored in all the
                    matches:\n\t" ) ;
            scanf ( "%f", &cp[i].avgrun ) ;
            fflush ( stdin ) ;
        }
    }
```

Files

[67] A hospital keeps a file of blood donors in which each record ha has the format:

Name: 20 Columns
Address: 40 Columns
Age: 2 Columns
Blood Type: 1 Columns (Type 1, 2, 3 or 4)

Write a program to read the file and print a list of all blood donors whose age is below 25 and blood is type 2.

Program:

```
/* To Read file in binary mode and display its contents */

#include <stdio.h>

main( )
{
    struct donors
    {
        char name[21] ;
        char address[41] ;
        int age ;
        char bloodtype ;
    } ;
    struct donors hospital ;
    FILE *fp ;

    fp = fopen ( "hospital.dat", "rb" ) ;  /* read only in binary mode */

    if ( fp == NULL )
    {
        puts ( "Cannot open file" ) ;
        exit( ) ;
    }

    while ( fread ( &hospital, sizeof ( hospital ), 1, fp ) == 1 )
    {
        if ( hospital.age < 25 && hospital.bloodtype == '2' )
            printf ( "\n%s %s %d %c", hospital.name, hospital.address,
                    hospital.age, hospital.bloodtype ) ;
    }
```



```
    fclose ( fp ) ;
}

/* In order to run this program, you must have a file "Hospital.dat". If you
   do not have such a file, you can use the following code to create this
   file with 5 records. */

#include <stdio.h>

main( )
{
    struct donors
    {
        char name[21] ;
        char address[41] ;
        int age ;
        char bloodtype ;
    } ;
    struct donors hospital ;
    int i ;
    FILE *fp ;

    fp = fopen ( "hospital.dat", "wb" ) ;
    if ( fp == NULL )
    {
        puts ( "Cannot open file" ) ;
        exit( ) ;
    }
    for ( i = 0 ; i < 5 ; i ++ )
    {
        printf ( "\n Name : " ) ;
        scanf ( "%s", hospital.name ) ;
        fflush ( stdin ) ;
        printf ( "\n add : " ) ;
        scanf ( "%s", hospital.address ) ;
        fflush ( stdin ) ;
        printf ( "\n age : " ) ;
        scanf ( "%d", &hospital.age ) ;
```


www.indianfun.org

```
        fflush ( stdin ) ;
        printf ( "\n type : " ) ;
        scanf ( "%c", &hospital.bloodtype ) ;
        fflush ( stdin ) ;
        fwrite ( &hospital, sizeof ( hospital ), 1, fp ) ;
    }
    fcloseall ( ) ;
}
```

[68] Given a list of names of students in a class. Write a program to store the names in a file on disk, display the **n**th name in the list (**n** is data to be read), display all names starting with S.

Program:

```
/* Create a file on disk, retrieve data as required */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *fp ;
    char  name[21], ch, another = 'y' ;
    int  num, n ;

    clrscr( ) ;

    fp = fopen ( "student.dat", "w+" ) ;  /* Open a file in write mode */

    if ( fp == NULL )
    {
        puts ( "Unable to create file" ) ;
        exit( ) ;
    }

    /* Loop for data entry */
```



```
    while ( another == 'y' || another == 'Y' )
    {
        puts ( "\nEnter the name of student: " ) ;
        gets ( name ) ;

        fputs ( name, fp ) ;  /* write data to file */
        fputs ( "\n", fp ) ;  /* add newline character at the end
                                 of record */

        puts ( "Do you want to add more names y/n" ) ;
        fflush ( stdin ) ;
        another = getch( ) ;
    }

    fseek ( fp, 0L, SEEK_SET ) ;  /* File pointer reset to start of the file */

    puts ( "\nEnter any number from the list" ) ;
    scanf ( "%d", &num ) ;
    n = num ;
    while ( fgets ( name, 21, fp ) != NULL )
    {
        num-- ;  /* count downwords to reach the required record */
        if ( num == 0 )
            printf ( "\nThe name of student no. %d is: %s\n", n, name ) ;
    }

    if ( num > 0 )
        puts ( "No such number exists in the list" ) ;

    fseek ( fp, 0L, SEEK_SET ) ;  /* File pointer reset again */

    puts ( "\nThe list of students whose name starts with S are : " ) ;

    while ( fgets ( name, 21, fp ) != NULL )
    {
        if ( name[0] == 's' || name[0] == 'S' )
            printf ( "%s", name ) ;
    }
```

```
    fclose ( fp ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}
```

[69] Assume that a Master file contains two fields, Roll no. and name of the student. At the end of the year, a set of students join the class and another set leaves. A Transaction file contains the roll numbers and an appropriate code to add or delete a student.

Write a program to create another file which contains the updated list names and roll numbers. Assume that the Master file and the Transaction file are arranged in ascending order by roll numbers. The updated file should also be in ascending order by roll numbers.

Program:

```
/* Create an updated file from Master & Transaction File */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *mf, *tf, *uf ;

    struct master  /* Master file structure */
    {
        int rollno ;
        char name[20] ;
    } ;

    struct transaction  /* Transaction file structure */
    {
        char code ;
        int rollup ;
        char name1[20] ;
    } ;

    struct master student, newstudent ;
    struct transaction add_del ;
    int msz = sizeof ( struct master ) ;

    clrscr( ) ;

    mf = fopen ( "master.dat", "r" ) ;  /* read only mode */

    if ( mf == NULL )
    {
        puts ( "Unable to open master file" ) ;
        exit( ) ;
    }

    tf = fopen ( "transact.dat", "r" ) ;  /* read only mode */

    if ( tf == NULL )
    {
        puts ( "Unable to open transaction file" ) ;
        fclose ( mf ) ;
        exit( ) ;
    }

    uf = fopen ( "updated.dat", "w+" ) ;  /* write and modify mode */

    if ( uf == NULL )
    {
        puts ( "Unable to create updated file" ) ;
        fclose ( mf ) ;
        fclose ( tf ) ;
        exit( ) ;
    }

    while ( fread ( &add_del, sizeof ( struct transaction ), 1, tf ) == 1 )
    {
```

```
        fread ( &student, msz, 1, mf ) ;

        /* if code = d, don't do anything */
        if ( add_del.code == 'd' )
        {
            if ( student.rollno == add_del.rollup )
                continue ;
            else
            {
                /* code != d, write to updated file from master file */
                while ( student.rollno != add_del.rollup )
                {
                    fwrite ( &student, msz, 1, uf ) ;
                    fread ( &student, msz, 1, mf ) ;
                }
            }
            continue ;
        }

        /* write all records from master file to updated file where roll no
           of master file < roll no of transaction file */
        while ( student.rollno < add_del.rollup )
        {
            fwrite ( &student, msz, 1, uf ) ;
            fread ( &student, msz, 1, mf ) ;
        }
        /* copy record from transaction file to updated file where
           code != d ( new student has to be added ) */
        newstudent.rollno = add_del.rollup ;
        strcpy ( newstudent.name, add_del.name1 ) ;

        fwrite ( &newstudent, msz, 1, uf ) ;

        fseek ( mf,-msz, SEEK_CUR ) ;

    } /*end of while loop*/

    /* write rest of the records from master file to updated file for which
       there is no matching record in transaction file */
    while ( fread ( &student, msz, 1, mf ) == 1 )
        fwrite ( &student, msz, 1, uf ) ;

    fclose ( mf ) ;
    fclose ( tf ) ;
    fclose ( uf ) ;

    printf ( "\nUpdated file saved. Press any key to exit..." ) ;
    getch( ) ;
}

/* The "master.dat" and "transact.dat" file can be created using
   the following code. Pl.ensure that your directory is correctly set. */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *mf, *tf, *uf ;

    struct master
    {
        int rollno ;
        char name[20] ;
    } ;

    struct transaction
    {
        char code ;
        int rollup ;
        char name1[20] ;
    } ;

    struct master student, newstudent ;
    struct transaction add_del ;
    int i, msz = sizeof ( struct master ) ;
```


www.indianfun.org

www.indianfun.org
www.indianfun.org

```
mf = fopen ( "master.dat", "w" ) ;

if ( mf == NULL )
{
    puts ( "Unable to open master file" ) ;
    exit( ) ;
}
for ( i = 0 ; i <= 2 ; i++ )
{
    printf ( "\n Rollno: " ) ;
    scanf ( "%d", &student.rollno ) ;
    fflush ( stdin ) ;
    printf ( "\n Name : " ) ;
    scanf ( "%s", &student.name ) ;
    fflush ( stdin ) ;
    fwrite ( &student, sizeof ( student ), 1, mf ) ;
}
fcloseall( ) ;

tf = fopen ( "transact.dat", "w" ) ;

if ( tf == NULL )
{
    puts ( "Unable to open master file" ) ;
    exit( ) ;
}
for ( i = 0 ; i <= 1 ; i++ )
{
    printf ( "\n Code : " ) ;
    scanf ( "%c", &add_del.code ) ;
    fflush ( stdin ) ;
    printf ( "\n Rollno : " ) ;
    scanf ( "%d", &add_del.rollup ) ;
    fflush ( stdin ) ;
    printf ( "\n Name :" ) ;
    scanf ( "%s", &add_del.name1 ) ;
    fflush ( stdin ) ;
    fwrite ( &add_del, sizeof ( add_del ), 1, tf ) ;
```



```
    }
    fcloseall( ) ;
}
```

[70] In a small firm employee numbers are given in serial numerical order, that is 1, 2, 3 etc.

(a) Create a file of employee data with following information: employee no, name, sex, gross salary.
(b) If more employees join, append their data to the file.
(c) If an employee with serial number 25 (say) leaves, delete the record by making gross salary 0.
(d) If some employee's gross salary increases, write a program to retrieve the record and update the salary.

Program:

```
/* Maintain An Employee Record in a small firm */

/* If the "emp.dat" file does not exist, this program can be used to create
   the file and add some records to it using the "add" choice. */

#include <stdio.h>
#include <conio.h>

main( )
{
    struct employee
    {
        int empno ;
        char name[20] ;
        char sex ;
        float gs ;
    } emp ;

    FILE *fp ;
    char another ;
```


www.indianfun.org

```
int delno, flag, choice ;
float newgs ;
long sz = sizeof ( struct employee ) ;

clrscr( ) ;

fp = fopen ( "emp.dat", "rb+" ) ; /* binary read / modify mode */
if ( fp == NULL )
{
    fp = fopen ( "emp.dat", "wb+") ; /* binary write / modify mode */
    if ( fp == NULL )
    {
        puts ( "Unable to open file" ) ;
        exit( ) ;
    }
}

/* Display Menu */
while ( 1 )
{
    clrscr( ) ;
    printf ( "1. Add\n" ) ;
    printf ( "2. Delete\n" ) ;
    printf ( "3. Update\n" ) ;
    printf ( "4. List\n" ) ;
    printf ( "0. Exit\n" ) ;

    printf ( "Enter choice: \n" ) ;
    scanf ( "%d", &choice ) ;

    clrscr( ) ;
    switch ( choice )  /* Take action based on choice */
    {
        case 1 :
            printf ( "Enter employee's number, name, sex and
                    gross salary: " ) ;
            scanf ( "%d %s %c %f", &emp.empno, emp.name,
                    &emp.sex, &emp.gs ) ;
```



```
            fwrite ( &emp, sz, 1, fp ) ; /* write new record to file */
        break ;

        case 2 :
            printf  ( "Enter employee number: " ) ;
            scanf ( "%d", &delno ) ;
            flag = 1 ;

            fseek ( fp, 0L, SEEK_SET ) ;
            while ( fread ( &emp, sz, 1, fp ) == 1 )
            {
                if ( emp.empno == delno )
                {
                    emp.gs = flag = 0 ;  /* employee deleted */

                    fseek ( fp, -sz, SEEK_CUR ) ;
                    fwrite ( &emp, sz, 1, fp ) ;  /* modify file */
                    break ;
                }
            }
            if ( flag )
                puts ( "No such employee number exists.\n" ) ;
        break ;

        case 3 :
            printf ( "Enter employee number and gross salary: " ) ;
            scanf ( "%d %f", &delno, &newgs ) ;
            flag = 1 ;

            fseek ( fp, 0L, SEEK_SET ) ;
            while ( fread ( &emp, sizeof ( emp ), 1, fp ) == 1 )
            {
                if ( emp.empno == delno )
                {
                    flag = 0 ;
                    emp.gs = newgs ;
                    fseek ( fp, -sz, SEEK_CUR ) ;
                    fwrite ( &emp, sz, 1, fp ) ;
```

```
                    break ;
                }
            }
            if ( flag )
                puts ( "No such employee number exists.\n" ) ;
            break ;

        case 4 :
            fseek ( fp, 0L, SEEK_SET ) ; /* List records from
                                            begining */
            while ( fread ( &emp, sizeof ( emp ), 1 ,fp ) == 1 )
                printf ( "%d %s %c %f\n",emp.empno,
                        emp.name,emp.sex,emp.gs ) ;
            break ;

        case 0 :
            fclose ( fp ) ;
            exit( ) ;
            break ;
    }
    printf ( "\n\n\nPress any key to continue..." ) ;
    getch( ) ;
  }
}
```

[71] Given a text file, create another text file deleting the words "a", "the", "an" and replacing each one of them with a blank space.

Program:

```
/* Create a new text file after replacing "a", "an" & "the" with a blank
   space from a given text file */

#include <stdio.h>
#include <string.h>
```



```
FILE *ft ;
int charcount ;

main( )
{
    FILE *fs ;
    char str[80], source[67], target[67] ;
    char *apstr[ ] =  {
                        "a",
                        "the",
                        "an",
                      } ;
    int i ;
    void newstr ( char *s, char *t ) ;

    clrscr( ) ;

    puts ( "Enter source file name : " ) ;
    gets ( source ) ;

    puts ( "Enter target file name : " ) ;
    gets ( target ) ;

    fs = fopen ( source, "r" ) ;  /* read only mode for source file */

    if ( fs == NULL )
    {
        puts ( "Unable to open source file." ) ;
        getch( ) ;
        exit( ) ;
    }

    ft = fopen ( target, "w+" ) ;  /* write / modify mode for target file */

    if ( ft == NULL )
    {
        fclose ( fs ) ;
        puts ( "Unable to create target file." ) ;
```


www.indianfun.org

```
        getch( ) ;
        exit( ) ;
    }

    while ( fgets ( str, 79, fs ) != NULL )
    {
        newstr ( str, apstr[0] ) ;

        for ( i = 1 ; i <= 2 ; i++ )
        {
            fseek ( ft, -charcount, SEEK_CUR ) ;
            fgets ( str, charcount, ft ) ;

            fseek ( ft, -charcount, SEEK_CUR ) ;
            newstr ( str, apstr[i] ) ;
        }

    }

    fclose ( fs ) ;
    fclose ( ft ) ;

    getch( ) ;
}

void newstr ( char *p, char *t )
{
    int len = strlen ( t ) ;
    charcount = 0 ;
    while ( *p )
    {
        if ( ( !strncmpi ( p, t, len ) && ( *( p - 1 ) == ' ' ) && ( *( p + len )
           == ' ' || *( p + len ) == '\n' ) ) || ( !strncmpi ( p, t, len ) && (
          charcount == 0 ) && ( *( p + len ) == ' ' || *( p + len ) == '\n' ) ) )
        {
            fputc ( ' ', ft ) ;
            p = p + strlen ( t ) - 1 ;
        }
```



```
        else
            fputc ( *p, ft ) ;
        p++ ;
        charcount++ ;
    }
    charcount++ ;
}
```

[72] You are given a data file EMPLOYEE.DAT with the following record structure:

```
struct employee {
                    int empno;
                    char name[30] ;
                    int basic, grade ;
                } ;
```

Every employee has a unique **empno** and there are no gaps. Records are entered into the data file in ascending order of employee number, **empno**. It is intended to check whether there are missing employee numbers. Write a program segment to read the data file records sequentially and display the list of missing employee numbers.

Program:

```
/* find missing employee numbers from a given file */

#include <stdio.h>
#include <conio.h>

main( )
{
    struct employee
    {
        int empno ;
        char name[30] ;
```

Ac 9028

www.indianfun.org

```
        int  basic, grade ;
    } ;
    struct employee emp ;
    FILE *fp ;
    int  count = 1 ;

    clrscr( ) ;

    fp = fopen ( "EMPLOYEE.DAT", "r") ;  /* Read only mode */

    if ( fp == NULL )
    {
        puts ( "Unable to open file." ) ;
        exit( ) ;
    }

    printf ( "The list of missing employee numbers:\n" ) ;

    while ( fread ( &emp, sizeof ( struct employee ), 1, fp ) == 1 )
    {
        if ( emp.empno != count )
        {
            for ( ; count < emp.empno ; count++ )
                printf ( "%d\n", count ) ;
        }
        count++ ;
    }

    fclose ( fp ) ;
    printf ( "\n\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;
}

/* The "Employee.dat" file can be created using the following code */
/* Pl. ensure that your directory is set correctly */

#include <stdio.h>
#include <conio.h>
```



```
main( )
{
    FILE *mf ;

    struct employee
    {
        int  empno ;
        char  name[30] ;
        int  basic, grade ;
    } ;

    struct employee emp ;
    int  i, msz = sizeof ( struct employee ) ;

    clrscr( ) ;

    mf = fopen ( "employee.dat", "w" ) ;

    if ( mf == NULL )
    {
        puts ( "Unable to open file" ) ;
        exit( ) ;
    }

    printf ( "\nEnter Values :" ) ;
    for ( i =  0 ; i <= 5 ; i++ )
    {
        printf ( "\n Employee No : " ) ;
        scanf ( "%d", &emp.empno ) ;
        fflush ( stdin ) ;
        printf ( "\n Name : " ) ;
        scanf ( "%s", &emp.name ) ;
        fflush ( stdin ) ;
        printf ( "\n Basic : " ) ;
        scanf ( "%d", &emp.basic ) ;
        printf ( "\n Grade : " ) ;
        scanf ( "%d", &emp.grade ) ;
        fflush ( stdin ) ;
```

```
        fwrite ( &emp, msz, 1, mf ) ;
    }
    fcloseall( ) ;
    printf ( "\nFile employee.dat has been created.. " ) ;
    getch( ) ;
}
```

[73] Write a program to carry out the following:

(a) to read a text file "TRIAL.TXT" consisting of a maximum of 50 lines of text, each line with a maximum of 80 characters.
(b) Count and display the number of words contained in the file.
(c) Display the total number of four letter words in the text file.

(Assume that the end of a word may be a space, comma or a fullstop followed by one or more spaces or a newline character)

Program:

```
/* count number of words in a file */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *fp ;
    char str[80] ;
    int words = 0, f_l_words = 0, i ;

    fp = fopen ( "TRIAL.TXT", "r" ) ;

    if ( fp == NULL )
    {
        puts ( "Unable to open file" ) ;
        exit( ) ;
    }

    while ( ( fgets ( str, 79, fp ) ) != NULL )
    {
        for ( i = 1 ; str[i] != '\0' ; i++ )
        {
            if ( str[i]== '.' || ( str[i] == '\n' && str[i-1] != '.' ) ||
                str[i] == ',' || ( str[ i ] == ' ' && str[i-1] != ',' &&
                str[i-1] != '.' ) )
                words++ ;

            if ( i < 74 )
            {
                if ( str[i] == ' ' && ( str[i + 5] == ' ' ||
                    str[i + 5] == '\n' || str[i + 5] == '.' ||
                    str[i + 5] == ',' ) )
                    f_l_words++ ;
                if ( i == 1 && ( str[4] == ' ' || str[4] == ',' || str[4] == '.' )
                    && ( str[1] != ' ' && str[1] != ',' && str[1] != '.' )
                    && ( str[2] != ' ' && str[2] != ',' && str[2] != '.' ) )
                    f_l_words++ ;
            }
        }
    }
    clrscr( ) ;

    printf ( "\nThe total number of words are %d.", words ) ;
    printf ( "\nThe number of four letter words are %d.", f_l_words ) ;
    fclose ( fp ) ;
}
```

[74] Write a program to read a list of words, sort the words in alphabetical order and display them one word per line. Also give the total number of words in the list. Output format should be:

Total Number of words in the list is ----------
Alphabetical listing of words

Assume the end of the list is indicated by ZZZZZZ and there are maximum in 25 words in the Text file.

Program:

```
/* Read, sort and display list of words from a file */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *fp ;
    char  *str[25], words[25][25], *t ;
    int  i, j, n ;

    clrscr( ) ;

    fp = fopen ( "trial.TXT", "r" ) ;

    if ( fp == NULL )
    {
        puts ( "Unable to open file." ) ;
        exit( ) ;
    }

    for ( n = 0 ; n <= 24 ; n++ )
        str[n] = words[n] ;

    for ( n = 0 ; fgets ( str[n], 24, fp ) != NULL ; n++ )
```



```
    for ( i = 0 ; i <= n - 2 ; i++ )
    {
        for ( j = i + 1 ; j <= n - 1 ; j++ )
        {
            if ( strcmp ( str[i], str[j] ) > 0 )
            {
                t = str[i] ;
                str[i] = str[j] ;
                str[j] = t ;
            }
        }
    }

    printf ( "Total Number of words in the list is %d\n", n ) ;
    printf ( "Alphabetical listing of words\n" ) ;

    for ( i = 0 ; i <= n - 1 ; i++ )
        printf ( "%s\n", str[i] ) ;

    fclose ( fp ) ;
    printf ( "\n\n\n\nPress any key to exit..." ) ;
    getch( ) ;

}
```

[75] Write a program to carry out the following:

(a) To read a text file 'INPUT.TXT'.
(b) Print each word in reverse order.

Note: Assume that each word length is maximum of 10 characters and each word is separated by newline / blank characters.

For example,

Input: INDIA IS MY COUNTRY
Output: AIDNI SI YM YRTNUOC

Program:

```
/* Read a text file and print each word in reverse order */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *fp ;
    char  str[11], ch ;
    int  i = 0 ;

    clrscr( ) ;

    fp = fopen ( "INPUT.TXT", "r" ) ;

    if ( fp == NULL )
    {
        puts ( "Unable to open file" ) ;
        exit( ) ;
    }

    while ( ( ch = getc ( fp ) ) != EOF )
    {
        if ( ch == '\n' || ch == ' ' )
        {
            str[i] = '\0' ;
            strrev ( str ) ;
            printf ( "%s ", str ) ;
            i = 0 ;
        }
        else
```



```
            str[i++] = ch
    }

    fclose ( fp ) ;
}
```

[76] Write a C program to read a large text file 'NOTES.TXT' and print it on the printer in cut-sheets, introducing page breaks at the end of every 50 lines and a pause message on the screen at the end of every page for the user to change the paper.

Program:

```
/* Reading a disk file and printing on cut sheets */

#include <stdio.h>
#include <conio.h>

main( )
{
    FILE *fp ;
    /* Counters for characters per line & number of lines per page */
    int charcount = 0, linecount = 0 ;
    char  ch ;

    fp = fopen ( "NOTES.TXT", "r" ) ; /* Read only mode */

    if ( fp == NULL )
    {
        puts ( "Unable to open file" ) ;
        exit( ) ;
    }

    /* Read until end of file */
    while ( ( ch = fgetc ( fp ) ) != EOF )
    {
        if ( ch == '\t' )
```

```
            charcount += 3 ;  /* Add 3 characters for a tab */

        charcount++ ;

        if ( ch == '\n' || charcount >= 80 )
        {
            linecount++ ;  /* Increment line counter after 80
                              characters */
            charcount = 0 ;  /* Reset character counter */
        }
        if ( linecount == 49 )  /* Print  49 lines per page */
        {
            puts ( "Insert another page, then press any key..." ) ;
            getch( ) ;
            linecount = 0 ;  /* Reset line counter */
        }

        fputc ( ch, stdprn ) ;  /* send to printer */
    }

    fclose ( fp ) ;
}
```

[77] A file contains a C program, but by mistake a novice has typed it out in capitals. It's a long program, therefore you cannot afford to have it retyped. Write a utility which reads this program and converts it to an appropriate small-case program and writes it to a target file. Use function **fgets( )** and **fput( )**.

Program:

```
/* Convert caps to small case using fgets() & fputs() */

#include <stdio.h>
#include <conio.h>
```



```
main( )
{
    FILE *fs, *ft ;
    char  str[80], target[67], source[67] ;
    int  i ;

    puts ( "Enter source file name: " ) ;
    gets ( source ) ;

    fs = fopen ( source, "r" ) ;   /* Read only mode */

    if ( fs == NULL )
    {
        puts ( "Unable to open source file" ) ;
        exit( ) ;
    }

    puts ( "Enter target file name: " ) ;
    gets ( target ) ;

    ft = fopen ( target, "w" ) ;  /* Write mode */

    if ( ft == NULL )
    {
        puts ( "Unable to create target file" ) ;
        fclose ( fs ) ;
        exit( ) ;
    }

    while ( fgets ( str, 79, fs ) != NULL )  /* Read from source file */
    {
        for ( i = 0 ; i <= 79 ; i++ )
            if ( str[i] >= 65 && str[i] <= 90 )
                str[i] += 32 ;  /* Convert caps to small case */
        fputs ( str, ft ) ;  /* Write to target file */
    }
    fclose ( fs ) ;
    fclose ( ft ) ;
```


www.indianfun.org

```
}
```

Miscellaneous

[78] A company pays normal ways for work during week days from Monday to Friday and 1.5 times wage for work on Saturday and Sunday. Given data in the following form:

Employee Number, Wage/Hour, hours worked on Monday, hors on Tuesday, ..., hours on Sunday.
Write a program to write out the Employee number and weekly wages. Use enumerated data type in your program.

Program:

```
/* Payroll using Enums */

#include <stdio.h>
#include <conio.h>
#include <math.h>

/* Function to link floating point  format */
fun( )
{
    float  f, *ff = &f ;
}

struct comp_pay
{
    int  emp_no ;
    float  wph ;
    int  days[7] ;
} ;

enum days
{
```



```
    mon, tue, wed, thu, fri, sat, sun
} ;

struct comp_pay cp[2] ;

main( )
{
    int  i, wday ;
    float  sum, wages = 0.0 ;
    char  *d_name[ ] = {
                        "Monday",
                        "Tuesday",
                        "Wednesday",
                        "Thursday",
                        "Friday",
                        "Saturday",
                        "Sunday"
                      } ;

    clrscr( ) ;

    printf ( "\nEnter the data for each employee:\n" ) ;
    for ( i = 0 ; i <= 1 ; i++ )
    {
        printf ( "\nEmployee Number = " ) ;
        scanf ( "%d", &cp[i].emp_no ) ;
        printf ( "Wages/hour  = " ) ;
        scanf ( "%f", &cp[i].wph ) ;
        for ( wday = mon ; wday <= sun ; wday++ )
        {
            printf ( "Total Hours worked on %s : ", d_name[wday] ) ;
            scanf ( "%d", &cp[i].days[wday] ) ;
        }
        clrscr( ) ;
    }
    clrscr( ) ;

    /* Calculate wages & print  the details */
```

```
for ( i = 0 ; i <= 1 ; i++ )
{
    printf ( "\n\nEmployee Number = %d", cp[i].emp_no ) ;
    printf ( "\nWages/hour  = %.2f", cp[i].wph ) ;
    for ( wday = mon ; wday <= sun ; wday++ )
    {

        printf ( "\nTotal hours worked on %s = %d", d_name[wday],
                cp[i].days[wday] ) ;

        if ( wday <= fri )
        {
            wages = wages + cp[i].days[wday] * cp[i].wph ;
        }
        else
        {
            if ( wday == sat && cp[i].days[sat] > 0 )
            {
                wages = wages + cp[i].days[sat] * ( cp[i].wph *
                        3.0 / 2 ) ;
            }
            if ( wday == sun && cp[i].days[sun] > 0 )
            {
                wages = wages + cp[i].days[sun] * cp[i].wph *
                        3.0 / 2 ;
            }
        }
    }
    printf ( "\n\nTotal wages/week = %f", wages ) ;
    wages = 0.0 ;
}
getch( ) ;
}
```

[79] With respect to the following program, say whether the subsequence remarks are true or false.

```
typedef struct {
        int month ;
        int day;
        int year ;
    } date;
struct {
    int acct_no ;
    char acct_type ;
    char name [80] ;
    float balance ;
    date lastpayment ;
} customer, *pc = & customer ;
```

(a) **customer.balance** and **(*pc).balance** refers to the same value

Answer: True

(b) **pc->lastpayment.month** is syntactically valid

Answer: True

(c) ***(pc->lastpayment.month)** is syntactically invalid

Answer: True

(d) ***(customer.name + 2)** and **customer.name[2]** both refer to **pc-> (name + 2)**

Answer: False

(e) ***((*pc).name + 2)** refer to the second character of customer's name

Answer: True

Program:

```
typedef struct
{
    int month ;
    int day ;
    int year ;
} date ;

struct {
        int acc_no ;
        char acct_type ;
        char name[80] ;
        float balance ;
        date lastpayment ;
      } customer, *pc = &customer ;

main( )
{
    clrscr( ) ;
    customer.acc_no = 1 ;
    customer.name[0] = 'a' ;
    customer.name[1] = 'b' ;
    customer.name[2] = 'c' ;
    customer.acct_type = 'a' ;
    customer.balance = 100.00 ;
    customer.lastpayment.month = 6 ;
    customer.lastpayment.day = 3 ;
    customer.lastpayment.year = 1990 ;
    printf ( "\n %f\t %f", customer.balance, (*pc).balance ) ;
    printf ( "\n %d", pc->lastpayment.month ) ;
    printf ( "\n %c\t%c\t%c", *(customer.name + 2 ),
             customer.name[2], *(pc->name + 2 ) ) ;
    printf ( "\n %c", *( (*pc).name + 2 ) ) ;
    getch( ) ;
}
```

[80] Unless typecasting is used the following programs would give an Error/warning. Indicate where and how you would use typecasting to avoid the Error/warning.

(a)
```
main( )
{
    int i = 32 ;
    float *p ;
    p = &i ; /* Error */
}
```

Program:

```
main( )
{
    int i = 32 ;
    float *p ;
    p = ( float * ) &i ; /* Typecasting */
    clrscr( ) ;
    printf ( "\n%d\t%x", i, &i ) ;
    printf ( "\n%d\t%x", *( int * ) p, p ) ;
    getch( ) ;
}
```

(b)
```
main( )
{
    char far *scr ;
    scr = 0xB8000000 ; /* Error */
    *scr = 'A' ;
}
```

Program:

```
main( )
{
    char far *scr ;
    scr = ( char far * ) 0xb8000000 ; /* Typecasting */
    clrscr( ) ;
```

```
    *scr = 'A' ;
    getch( ) ;
}
```

(c) main( )

```
    {
        struct a
        {
            int i ;
            char c ;
        } ;
        struct a car 1 = { 10, 'A' }, var2, *ptr ;
        char ch ;

        ptr = &var| ;
        var2 = *ptr ; /* typecasting not required */

        /* one way accessing 'A' */
        ch = ptr -> c ;
        printf ( "n%c", ch ) ;
        /* another way of accessing 'A' */
        ch = * ( ptr + 2 ) ; /* Error */
        print  ( "\n%c", ch ) ;
    }
```

Program:

```
main( )
{
    struct a
    {
        int i ;
        char c ;
    } ;
    struct a var1 = { 10, 'A' }, var2, *ptr ;
    char ch ;
```



```
    clrscr( ) ;
    ptr = &var1 ;
    var2 = *ptr ;

    /*one way of accessing 'A'*/
    ch = ptr->c ;
    printf ( "\n%c", ch ) ;

    /*another way of accessing 'A' */
    ch = *( ( char  * )ptr + 2  ) ;  /* Error corrected */
    printf ( "\n%c", ch ) ;
    getch( ) ;
}
```

(d) main( )

```
    {
        struct a
        {
            int i ;
            char ch ;
        } ;
        struct a var 1 = { 10, 'A' } var2 ;
        char  *ptr ;

        ptr = ( char  * ) &var1 ;
        var2 = *(ptr ) ;  /* Error */
        printf ( "\n%d %c", var2.i, var2.ch ) ;
    }
```

Program:

```
main( )
{
    struct a
    {
        int i ;
        char ch ;
```

```
    } ;
    struct a var1 = { 10, 'A' }, var2 ;
    char  *ptr ;

    clrscr( ) ;
    ptr = ( char  * ) &var1 ;

    var2 = *( ( struct a * ) ptr ) ;  /* Corrected code */
    printf ( "\n%d\t%c", var2.i, var2.ch ) ;
    getch( ) ;
}
```

[81] Point out the Error *if* any in the following program. If there is No Error, what would be the Output of the program.

```
main( )
{
    struct a
    {
        int  i ;
        char  ch ;
    } ;
    struct a var1 = { 10, 'A' }, *ptr1, *ptr2 ;
    void **pp ;

    ptr1 = &var1 ;
    pp = ( void ** ) & ptr1 ;
    ptr2 = * ( ( struct a ** ) pp ) ;
    printf ( "\n%d %c", ptr2->I, ptr2->ch ) ;
}
```

Program:

```
/* OUTPUT of the program will be 10A */
```



```
main( )
{
    struct a
    {
        int  i ;
        char  ch ;
    } ;
    struct a var1 = { 10, 'A' }, *ptr1 , *ptr2 ;
    void **pp ;

    clrscr( ) ;
    ptr1 = &var1 ;
    pp = ( void ** ) &ptr1 ;
    ptr2 = *( ( struct a ** ) pp ) ;
    printf ( "\n%d%c", ptr2->i, ptr2->ch ) ;
    getch( ) ;
}
```