YENEPOYA INSTITUTE OF TECHNOLOGY

N.H.13, Thodar, Moodbidri, Mangalore, Karnataka 574225

(Affiliated of Visvesvaraya Technological University)

LABORATORY MANUAL

C PROGRAMMING LABORATORY

18CPL17/27

(Effective from the academic year 2018 -2019)

SEMESTER - I /II

Prepared By:

Prof. ANSER PASHA C.A.

Dept. of ISE, YIT

COMPUTER PROGRAMMING LABORATORY (Effective from the academic year 2018 -2019) SEMESTER – I/II

Subject Code 18CPL17/27	**CIE Marks** 40
Number of Contact Hours/Week 0:0:2	**SEE Marks** 60
Total Number of Lab Contact Hours 30	**Exam Hours** 3 Hrs

Credits – 1

Descriptions (if any):

- The laboratory should be preceded or followed by a tutorial to explain the approach or algorithm being implemented / implemented for the problems given.
- Note that experiment 1 is mandatory and written in the journal.
- Questions related with experiment 1, need to be asked during viva-voce for all experiments.
- Every experiment should have algorithm and flowchart be written before writing the program.
- Code should be traced using minimum two test cases which should be recorded.
- It is preferred to implement using Linux and GCC.

Laboratory Programs:
1. Familiarization with computer hardware and programming environment, concept of naming the program files, storing, compilation, execution and debugging. Taking any simple C- code.
PART A
2. Develop a program to solve simple computational problems using arithmetic expressions and use of each operator leading to simulation of a commercial calculator. (No built-in math function)
3. Develop a program to compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.
4. Develop a program to find the reverse of a positive integer and check for palindrome or not. Display appropriate messages.
5. An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.
6. Introduce 1D Array manipulation and implement Binary search.
7. Implement using functions to check whether the given number is prime and display appropriate messages. (No built-in math function)
PART B
8. Develop a program to introduce 2D Array manipulation and implement Matrix multiplication, and ensure the rules of multiplication are checked.
9. Develop a Program to compute Sin(x) using Taylor series approximation .Compare your result with the built- in Library function. Print both the results with appropriate messages.
10. Write functions to implement string operations such as compare, concatenate, string length. Convince the parameter passing techniques.
11. Develop a program to sort the given set of N numbers using Bubble sort.
12. Develop a program to find the square root of a given number N and execute for all possible inputs with appropriate messages. Note: Don't use library function sqrt(n).
13. Implement structures to read, write, and compute average- marks and the students scoring above and below the average marks for a class of N students.
14. Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.
15. Implement Recursive functions for Binary to Decimal Conversion

.

1. **Familiarization with computer hardware and programming environment, concept of naming the program files, storing, compilation, execution and debugging. Taking any simple C- code.**

Familiarization with computer hardware

A computer is an electronic device that processes data, converting it into information that is useful to people. Any computer regardless of its type is controlled by programmed instructions, which give the machine a purpose and tell it what to do.
A computer performs basically five major functions as shown in **Figure 1.1** they are:

1. It accepts data or instructions by the user in the way of input.
2. It stores the data.
3. It can processes the data as required by the user.
4. It gives result in the form of output.
5. It controls all operations inside a computer.

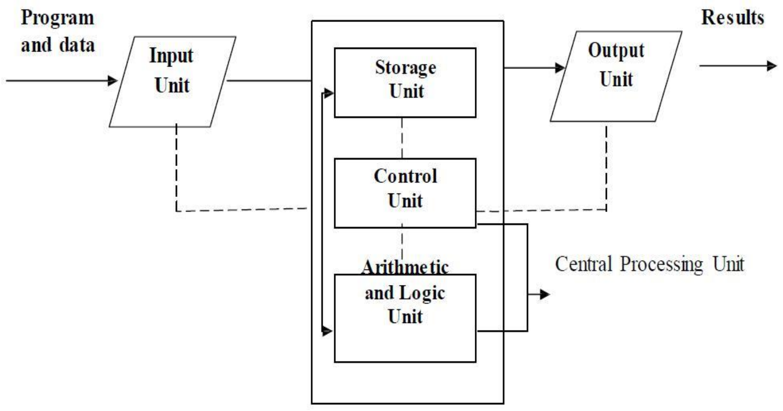


Figure 1.1: Functional block diagram of computer.

1. **Input:** This is the process of entering data and programs in to the computer system. It takes as inputs raw data and performs some processing giving out processed data. Standard input device is keyboard.
2. **Storage:** The process of saving data and instructions permanently is known as storage. Data has to be fed into the system before the actual processing starts. It is because the processing speed of Central Processing Unit (CPU) is so fast that the data has to be provided to CPU with the same speed. Therefore the data is first stored in the storage unit for faster access and processing. This storage unit or the primary storage of the computer system is designed to do the above functionality. It provides space for storing data and instructions.
   The storage unit performs the following major functions:
   - All data and instructions are stored here before and after processing.
   - Intermediate results of processing are also stored here.
3. **Processing:** The task of performing operations like arithmetic and logical operations is called processing. The Central Processing Unit (CPU) takes data and instructions from the storage unit and makes all sorts of calculations based on the instructions given and the type of data provided. It is then sent back to the storage unit.
4. **Output**: This is the process of producing results from the data for getting useful information. Similarly the output produced by the computer after processing must also be kept somewhere inside the computer before being given to you in human readable form. Again the output is also stored inside the computer for further processing.

5. **Control:** The manner how instructions are executed and the above operations are performed. Controlling of all operations like input, processing and output are performed by control unit. It takes care of step by step processing of all operations inside the computer.

Functional units

In order to carry out the operations mentioned in the previous section the computer allocates the task between its various functional units. The computer system is divided into three separate units for its operation. They are

1. Arithmetic logical unit
2. Control unit.
3. Central processing unit.

1. **Arithmetic Logical Unit (ALU)**
   After you enter data through the input device it is stored in the primary storage unit. The actual processing of the data and instruction are performed by Arithmetic Logical Unit. The major operations performed by the ALU are addition, subtraction, multiplication, division, logic and comparison. Data is transferred to ALU from storage unit when required. After processing the output is returned back to storage unit for further processing or getting stored.

2. **Control Unit (CU)**
   The next component of computer is the Control Unit, which acts like the supervisor seeing that things are done in proper fashion. Control Unit is responsible for coordinating various operations using time signal. The control unit determines the sequence in which computer programs and instructions are executed. Things like processing of programs stored in the main memory interpretation of the instructions and issuing of signals for other units of the computer to execute them. It also acts as a switch board operator when several users access the computer simultaneously. Thereby it coordinates the activities of computer's peripheral equipment as they perform the input and output.

3. **Central Processing Unit (CPU)**
   The ALU and the CU of a computer system are jointly known as the central processing unit. You may call CPU as the brain of any computer system. It is just like brain that takes all major decisions, makes all sorts of calculations and directs different parts of the computer functions by activating and controlling the operations.

How to Compile and Run a C Program on Linux

This document shows how to compile and run a C program on Linux using the gcc compiler.

Step 1. Open up a terminal

Search for the terminal application in the Dash tool (located as the topmost item in the Launcher).

Open up a terminal by clicking on the icon.

Step 2. Use a text editor to create the C source code.

Type the command

```
$gedit  hello.c
```

and enter the C source code below:

```
#include <stdio.h>
void main()
{
     printf("Hello World\n");
}
```

Close the editor window.

Step 3. Compile the program.

Type the command

```
$cc hello.c
```

This command will invoke the C compiler to compile the file hello.c and output the result to an executable called hello.

Step 4. Execute the program.

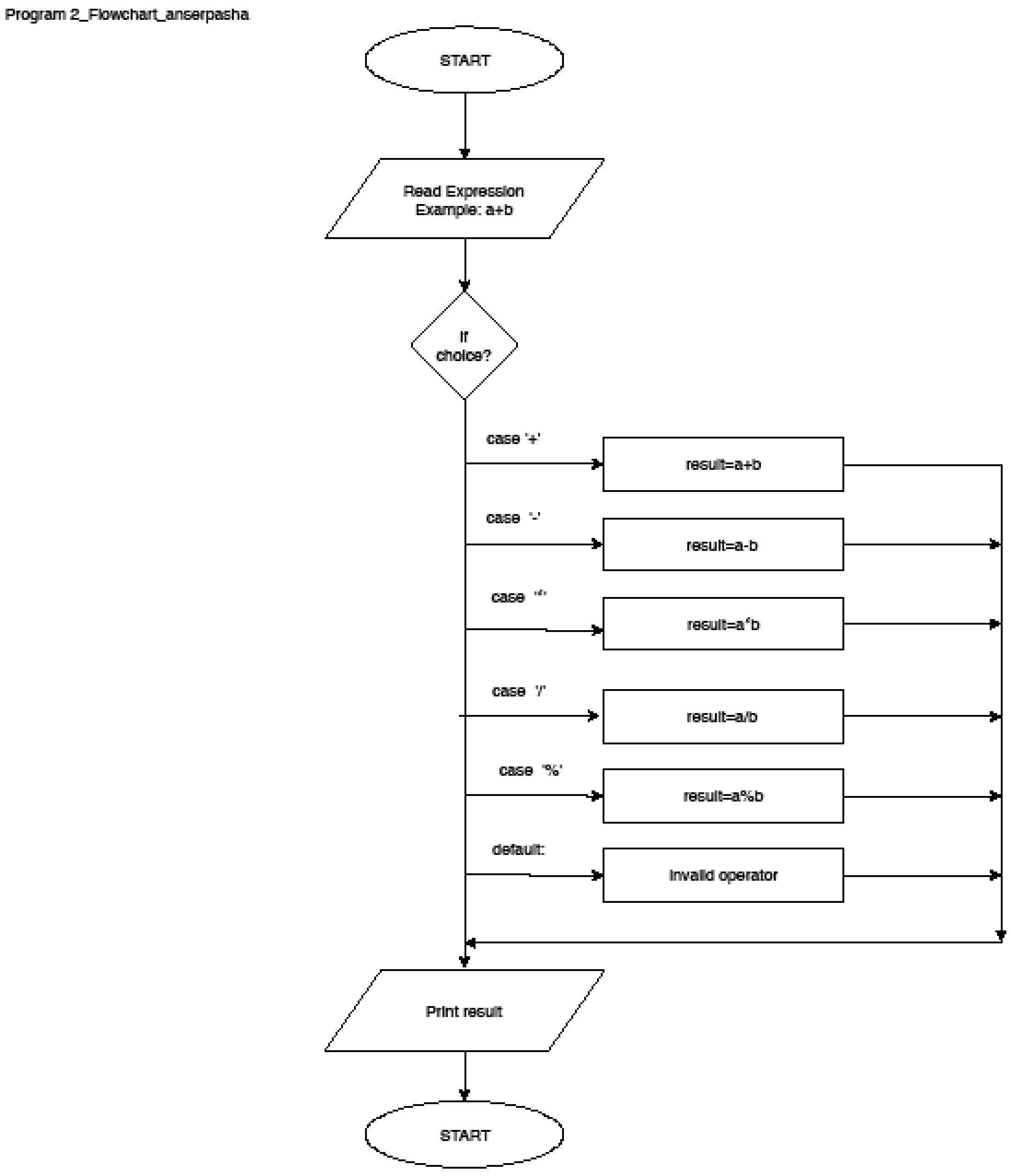
Type the command

```
./a.out
```

This should result in the output

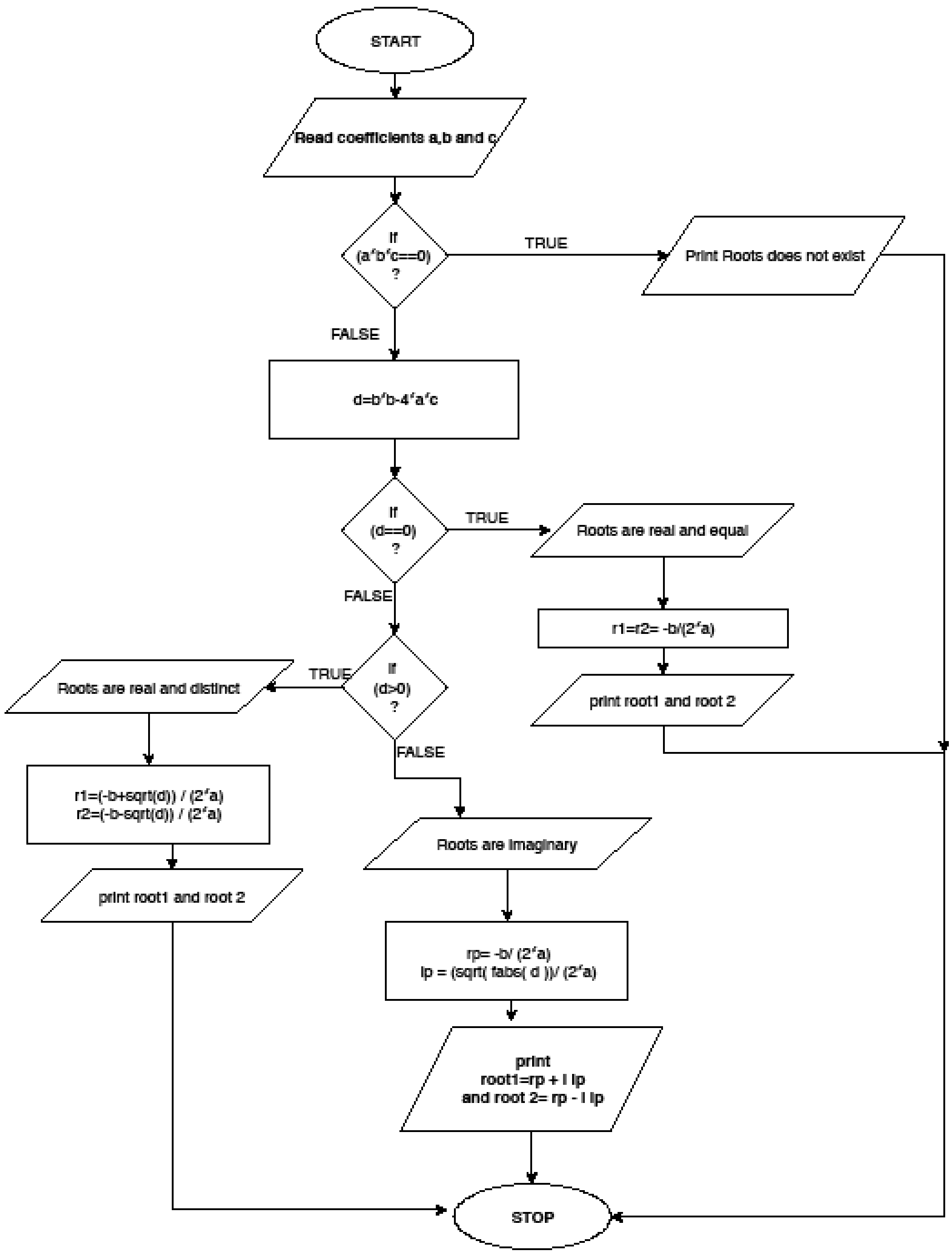
```
Hello World
```

2. Develop a program to solve simple computational problems using arithmetic expressions and use of each operator leading to simulation of a commercial calculator. (No built-in math function)



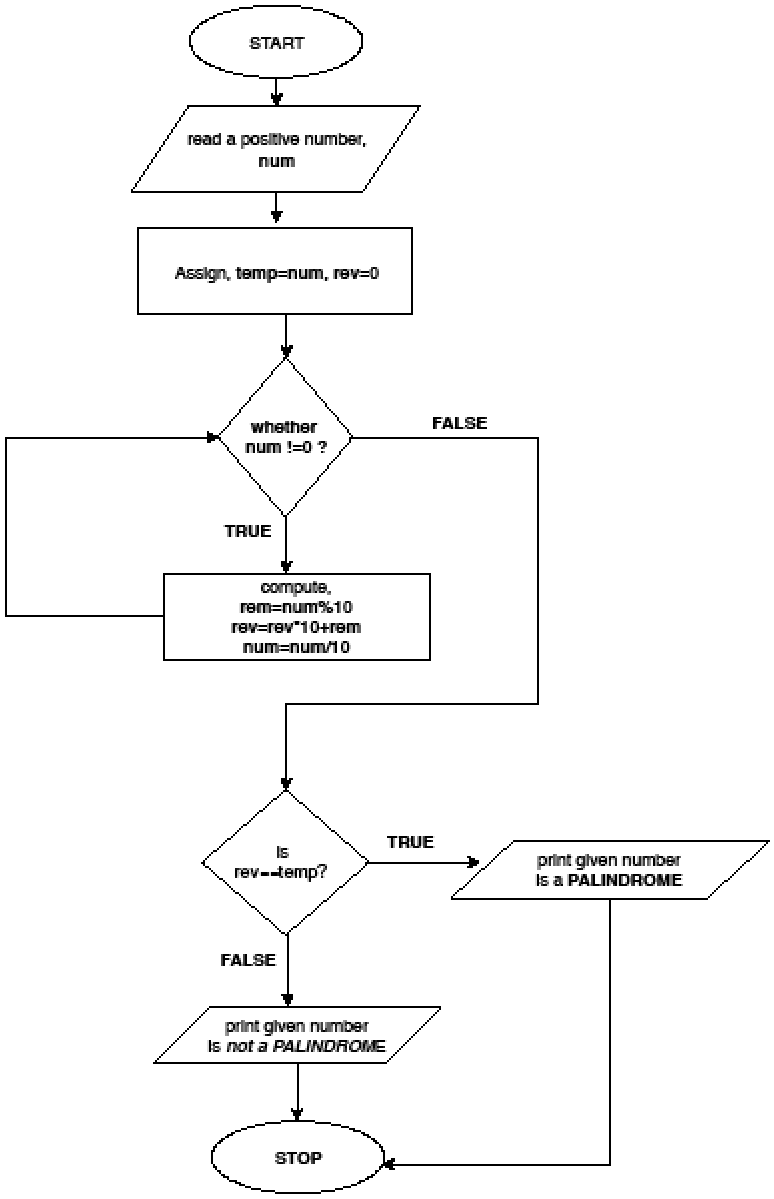
```
#include<stdio.h>

void main()
{
    int  a, b, result;
    char op;

printf("\n Enter arithmetic expression [operand1 operator operand2] :");
scanf("%d %c %d",&a, &op, &b);
switch(op)
{
    case  '+':          result =a+b;
                        printf("\n Sum = %d", result);
                        break;

    case  '-':          result =a-b;
                         printf("\n Difference = %d", result);
                        break;

    case  '*':          result =a*b;
                        printf("\n Product  = %d", result);
                        break;

    case   '/':         if(b!=0)
                        {
                                result=a/b;
                                printf("\n Quotient =  %d", result);
                        }
                        else
                    {
                               printf("\n Divide error!");
                        }
                        break;

    case  '%':          result =(a % b);
                        printf("\n Reminder =  %d", result);
                        break;

    default:            printf("\a Invalid operator!");
                        break;
   }

}
```

3. Develop a program to compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.



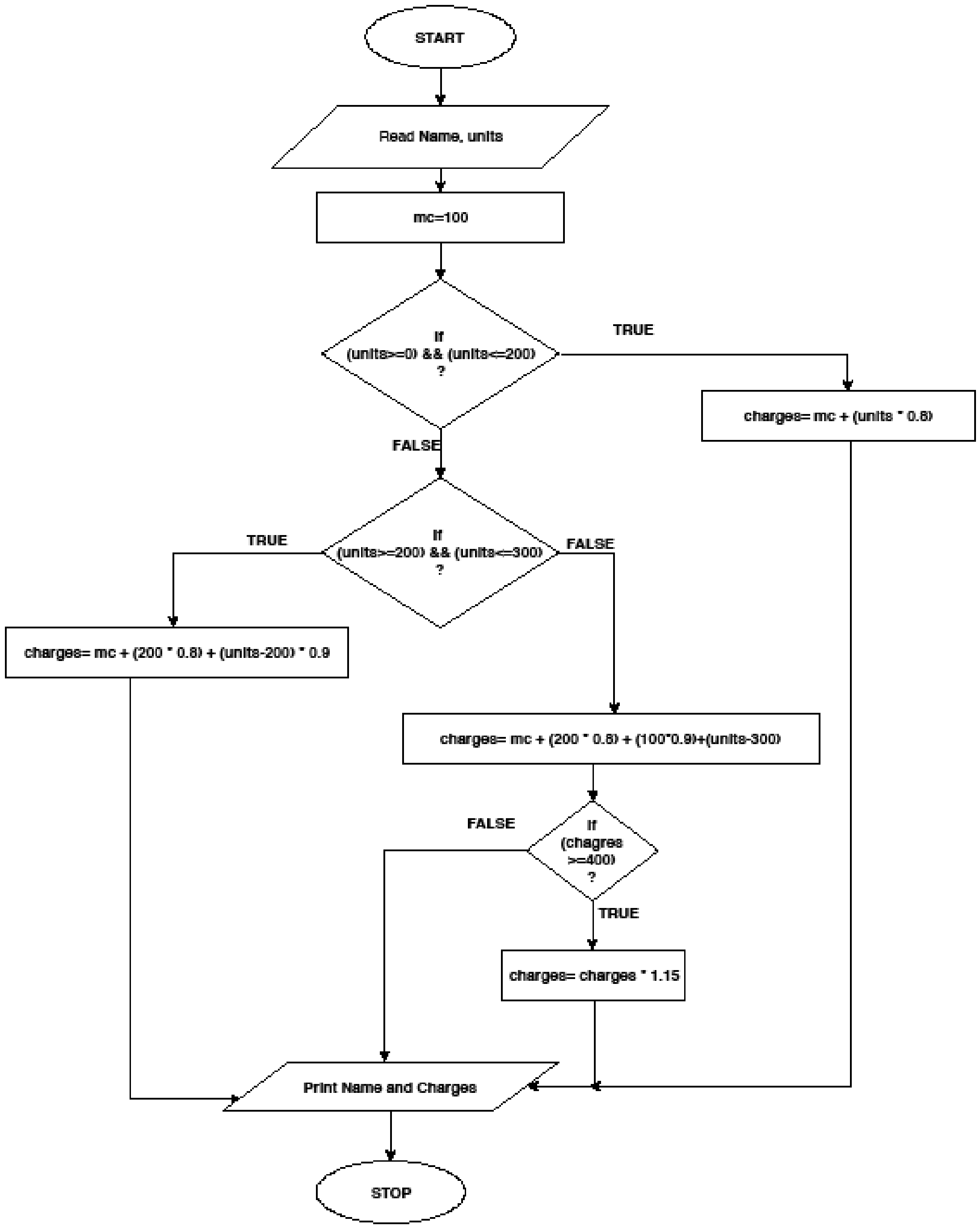
```
#include<stdio.h>
#include<math.h>

void main()
{

   float a,b,c, root1, root2, realp, imgp, disc;

   printf("\n\n Enter the values for coefficients a,b and c: ");
   scanf("%f%f%f",&a,&b,&c);
   if(a *b*c = = 0)
   {
      printf("\n Invalid Inputs  Try Again");
      return (0);
   }

   disc = b*b-4*a*c;

   if(disc = = 0)
   {
      printf("\nThe Roots are Equal");
      root1 = root2 = -b / (2.0*a);
      printf("\n Root1 = Root2 = %f", root1);
   }

   if(disc > 0)
   {
      printf("\n The Roots are Real and Distinct");
      root1 = (-b + sqrt(disc))/(2.0*a);
      root2 = (-b - sqrt(disc))/(2.0*a);
      printf("\nRoot1 = %f", root1);
      printf("\nRoot2 = %f", root2);
   }

   if(disc<0)
   {
      printf("\n The Roots are Imaginary");
      realp = -b/(2.0*a);
      imgp = sqrt(fabs(disc))/(2.0*a);  // fabs : absolute value of a Floating point Number
      printf("\nRoot1=%f +i %f",realp,imgp);
      printf("\nRoot2 = %f -i %f",realp,imgp);
   }
}
```

4. Develop a program to find the reverse of a positive integer and check for palindrome or not. Display appropriate messages.


START
read a positive number, num
Assign, temp=num, rev=0
whether num !=0 ?
FALSE
TRUE
compute, rem=num%10 rev=rev*10+rem num=num/10
Is rev==temp?
TRUE
print given number Is a PALINDROME
FALSE
print given number Is not a PALINDROME
STOP

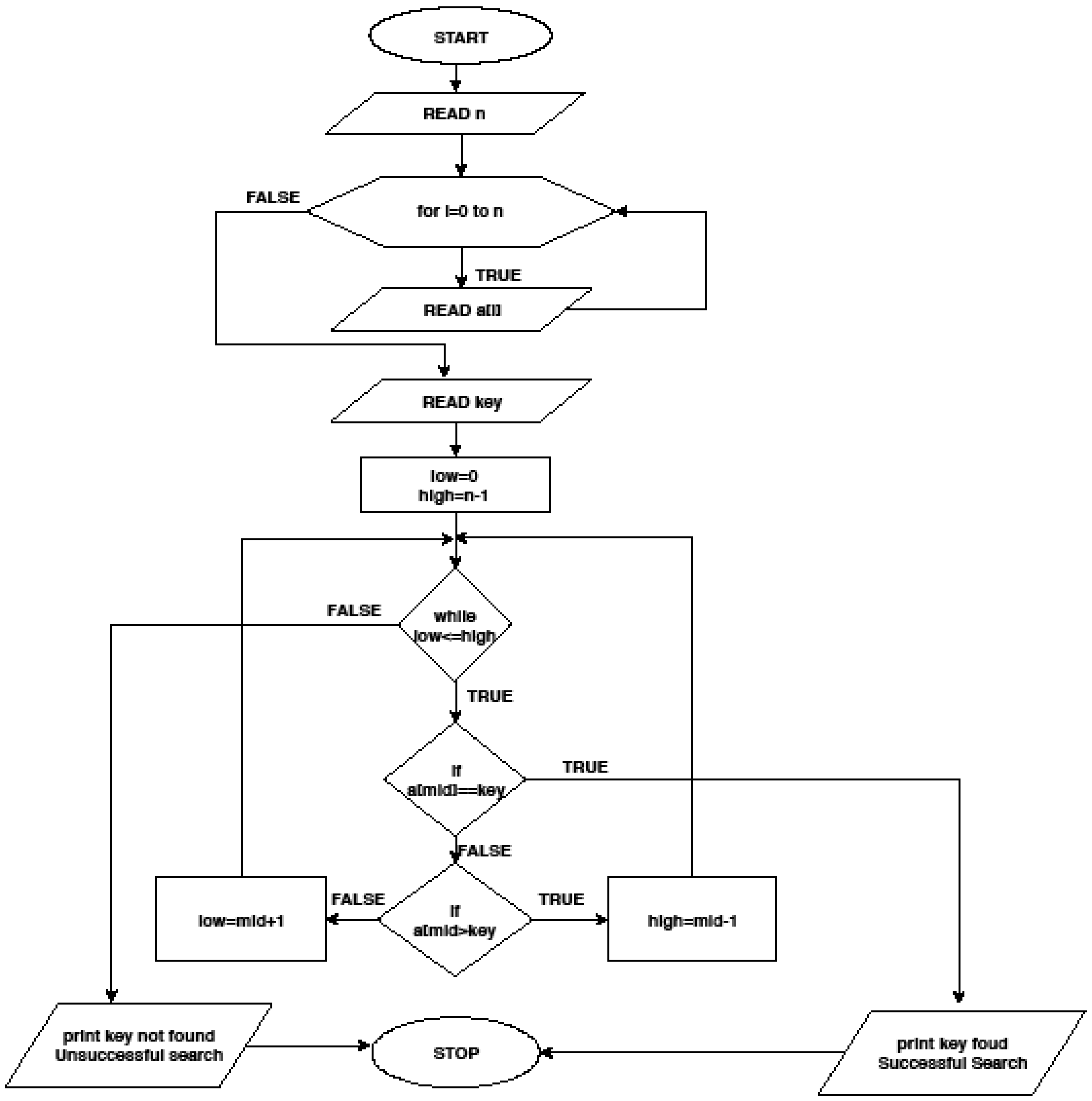
```
#include<stdio.h>

void main()
{

      int  num, temp, rev=0, rem;

      printf("\nEnter a valid integer number: ");
      scanf("%d",&num);

      if (num > 0)
      {
        temp = num;
        while(num != 0)
                {
                        rem = num % 10;
                        rev = rev * 10 + rem;
                        num = num / 10;
                }

                printf("\n The reversed number of  %d  is  %d",temp,rev);
                if(temp == rev)
                printf("\n %d is Palindrome\n",temp);
                else
                printf("\n The number is not palindrome\n");
     }
                else
                printf("\nInvalid Number\n");

  }
```

5. An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.


START
Read Name, units
mc=100
If (units>=0) && (units<=200) ?
TRUE
FALSE
charges= mc + (units * 0.8)
If (units>=200) && (units<=300) ?
TRUE
FALSE
charges= mc + (200 * 0.8) + (units-200) * 0.9
charges= mc + (200 * 0.8) + (100*0.9)+(units-300)
FALSE
If (chagres >=400) ?
TRUE
charges= charges * 1.15
Print Name and Charges
STOP

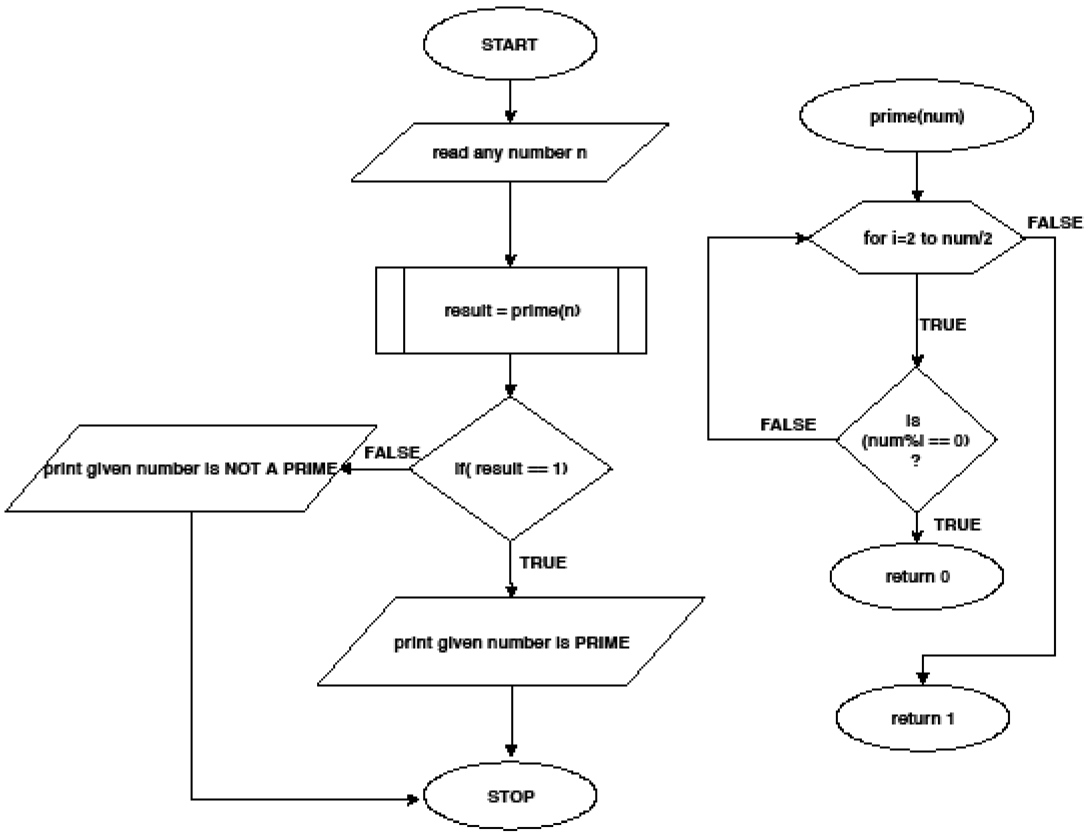
```
#include<stdio.h>
void main()
{
    char name[50];
    float charges,units,mc=100,sc=0,total;
    printf("\nEnter the Name of customer and number of units consumed\n");
    scanf("%s%f",&name,&units);
    if( units>=0 && units<=200)
    charges = ( units * 0.8 );
    else
    if(units>200 && units<=300)
    charges = ( 200 * 0.8 ) + ( units – 200 ) * 0.9;
    else
    charges = ( 200 * 0.8 ) + ( 100 * 0.9 ) + ( units – 300 );
    total = mc + charges;
    if(total > 400)
    {
    sc = total * 0.15;
    total = total + sc;
    }
        printf("\n**********ELECTRICITY BILL**********\n");
        printf("\n Customer Name : %s",name);
        printf("\n Units consumed : %f",units);
        printf("\n Meter charge : %f",mc);
        printf("\n charge for consumption : %f",charges);
        printf("\n Surcharge : %f",sc);
        printf("\n Total amount to pay : %f",total);
        printf("\n********************************\n\n");
}
```

6. Introduce 1D Array manipulation and implement Binary search.



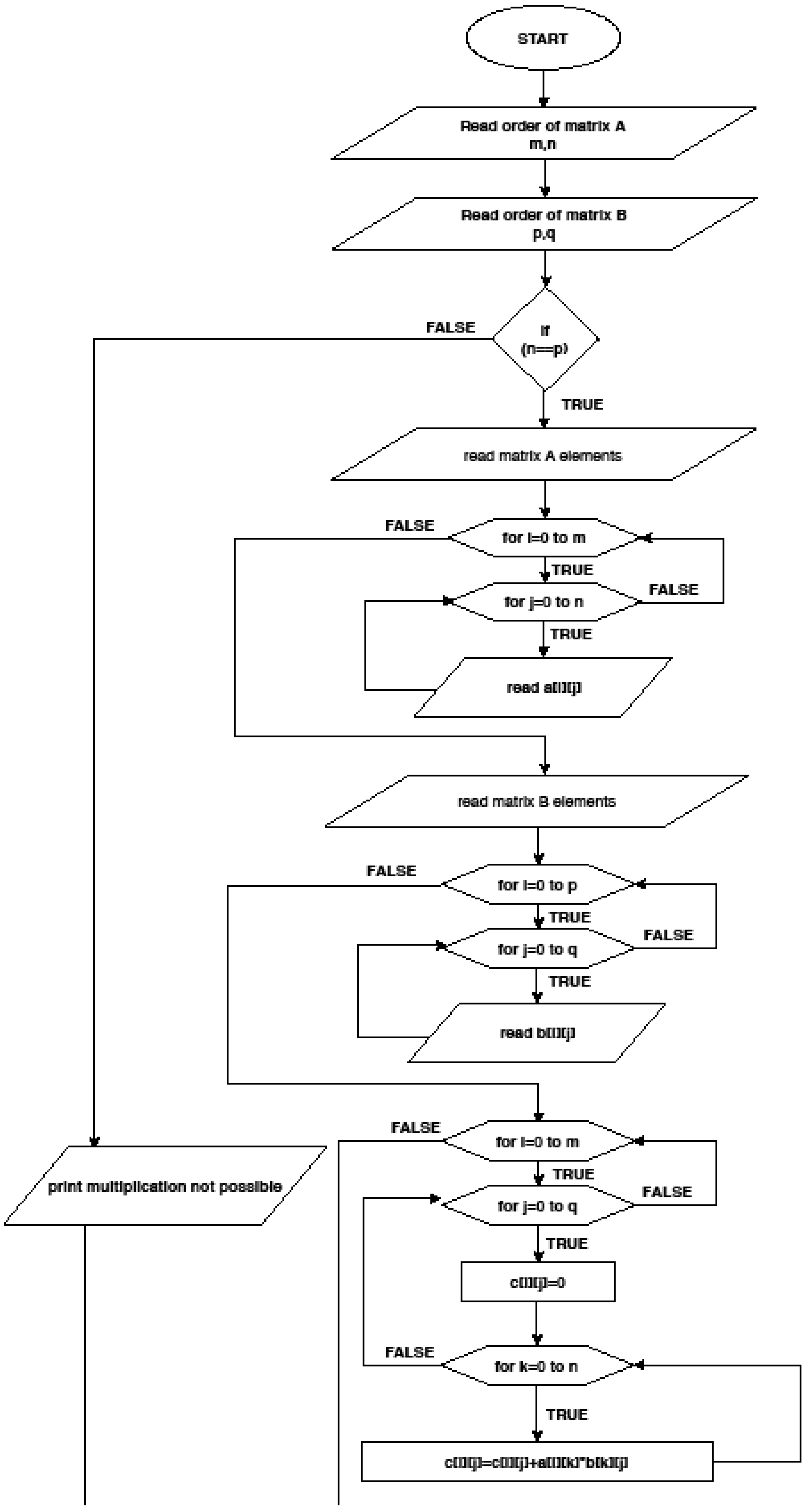
```
#include <stdio.h>

void main()
{
  int i, low, high, mid, n, key, arr[100],flag=0;
  printf("Enter number of elements\n");
  scanf("%d",&n);
  printf("Enter integer numbers in ascending order\n");

  for (i = 0; i < n; i++)
    scanf("%d",&arr[i]);

  printf("Enter value to Search\n");
  scanf("%d", &key);

  low = 0;
  high = n - 1;

  while (low <= high)
  {
      mid = (low+high)/2;

     if (key == arr[mid])
      {
       flag=1;
       break;
      }

    if (key > arr[mid] )
        low = mid + 1;

    if (key < arr[mid])
        high = mid - 1;
   }
   if ( flag==1)
         printf("%d found at location %d.\n", key, mid+1);
    else
        printf(" %d is Not found! \n", key);

}
```

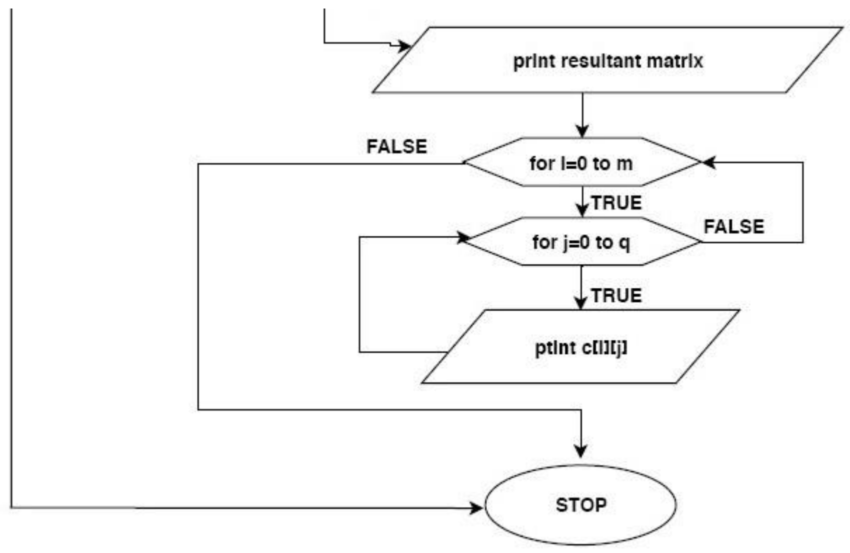
7. Implement using functions to check whether the given number is prime and display appropriate messages. (No built-in math function)



```
# include<stdio.h>

// Function definition
int isprime(int num)
{
    int i;
    for(i = 2  ;  i<= num / 2 ;  i++)
    {
        if ( num % i == 0)
            return 0;                    // not a prime
     }
    return 1;                            // a prime
}

void main()
{
    int  n,res;
    printf("\n Enter the integer numbers \n");
    scanf("%d",&n);
    res = isprime(n);    // function call
    if ( res == 1)
    printf("\nGiven number is Prime number\n");
    else
    printf("\nGiven number is not a Prime number\n");
}
```

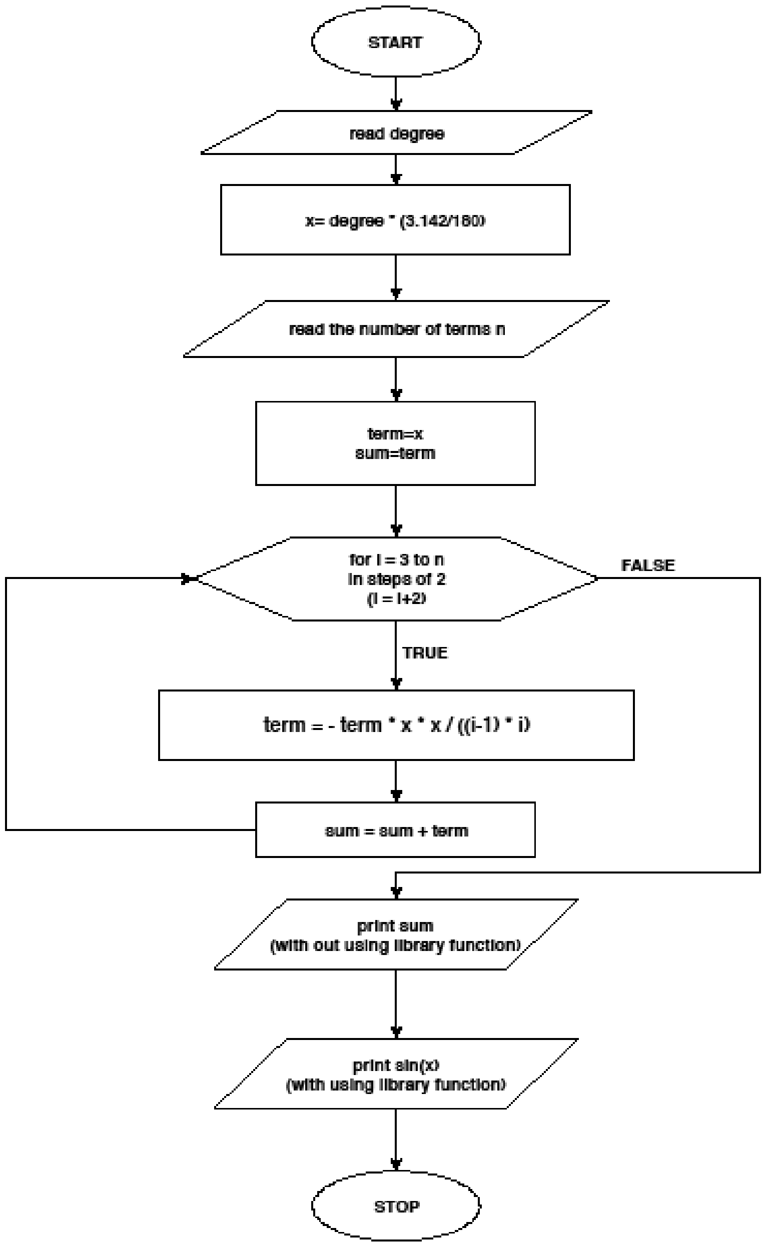
8. Develop a program to introduce 2D Array manipulation and implement Matrix multiplication and ensure the rules of multiplication are checked.




print resultant matrix
FALSE
for i=0 to m
TRUE
for j=0 to q
FALSE
TRUE
ptint c[i][j]
STOP

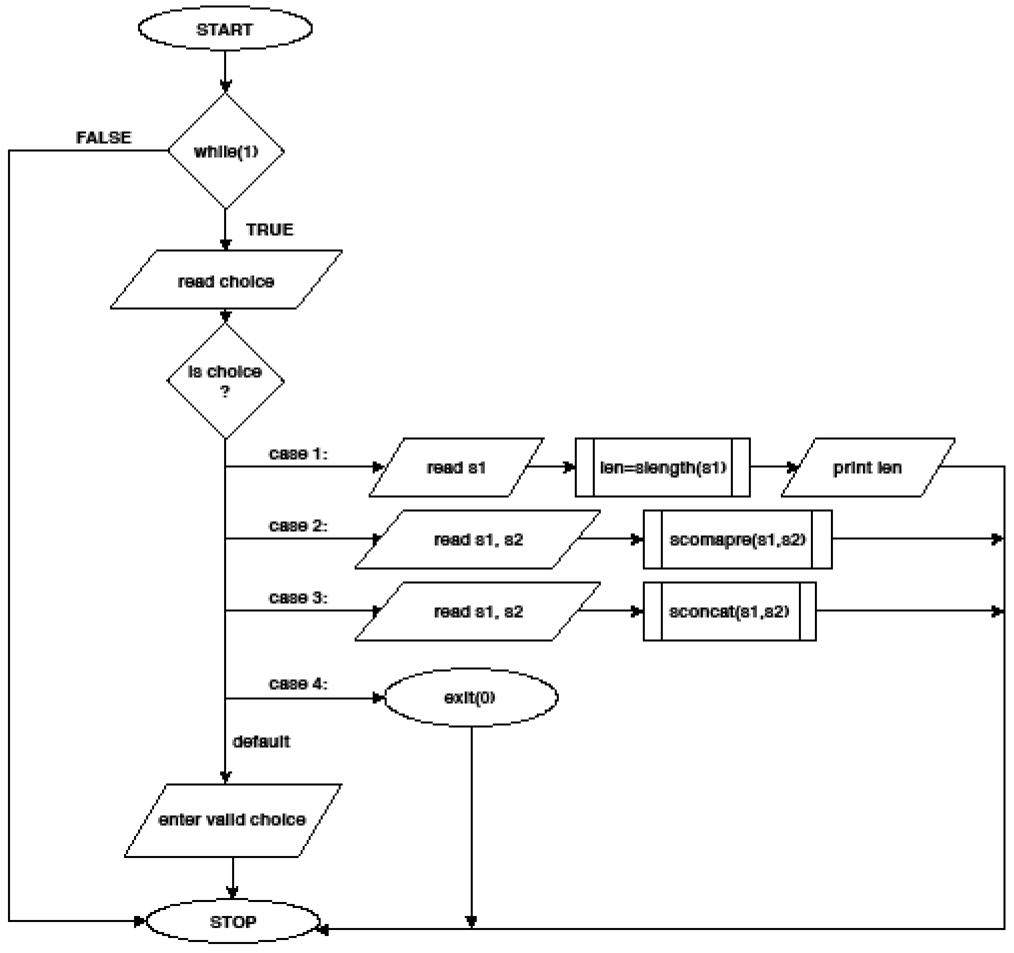
```
#include<stdio.h>

void main()
{

      int a[10][10],b[10][10],c[10][10],i,j,k,m,n,p,q;
      printf("Enter the order of matrix a : ");
      scanf("%d%d",&m,&n);

      printf("Enter the order of matrix b : ");
      scanf("%d%d",&p,&q);

      if(n!=p)
                printf("\nMultiplication not possible");

      else
      {
                  printf("\n Enter the elements of Matrix a :\n");
                  for(i=0;i<m;i++)
                              for(j=0;j<n;j++)
                                          scanf("%d",&a[i][j]);

            printf("Enter the elements of Matrix b :\n");
                  for(i=0;i<p;i++)
                              for(j=0;j<q;j++)
                                          scanf("%d",&b[j][i]);

                  for(i=0;i<m;i++)
                  for(j=0;j<q;j++)
                  {
                              c[i][j]=0;
                              for(k=0;k<n;k++)
                                          c[i][j]=c[i][j]+(a[i][k]*b[k][j]);
                  }

                  printf("\nThe Resultant Matrix is:\n");
                  for(i=0;i<m;i++)
                  {
                              for(j=0;j<q;j++)
                              {
                                          printf("%d\t",c[i][j]);
                              }
                              printf("\n");
          }
     }

}
```

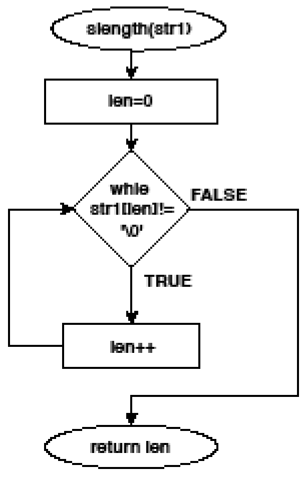
9. Develop a Program to compute Sin(x) using Taylor series approximation .Compare your result with the built- in Library function. Print both the results with appropriate messages.

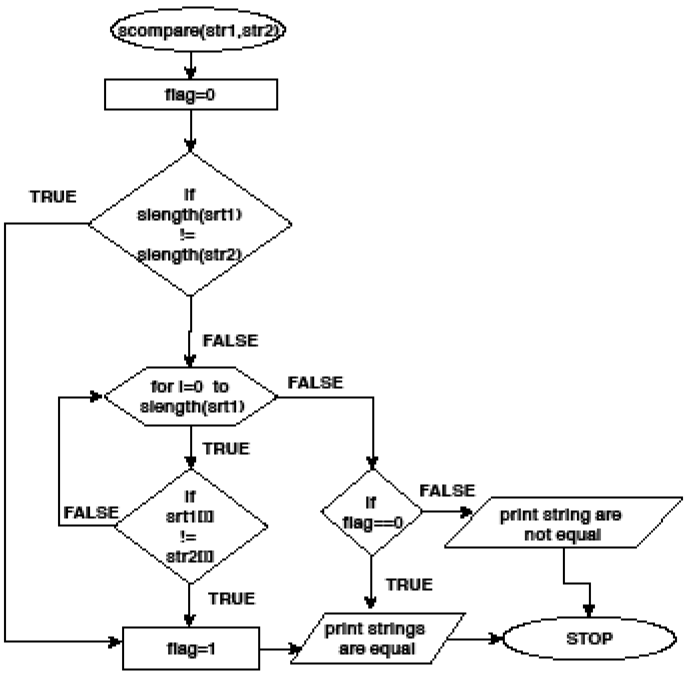


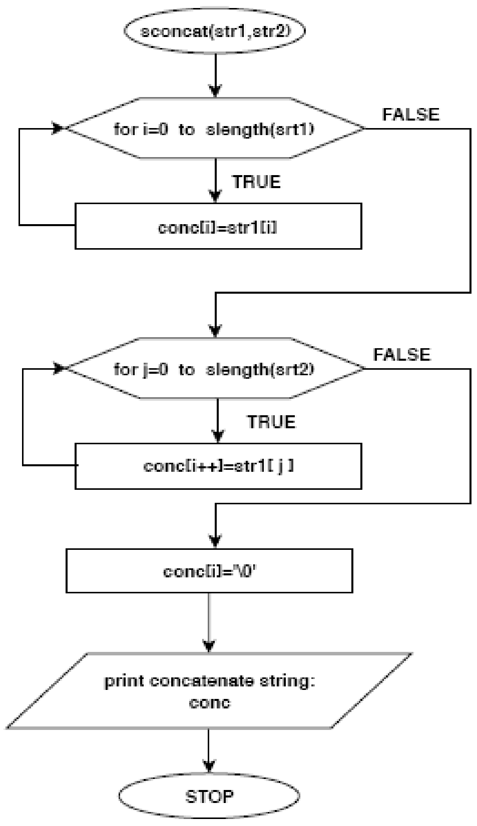
```
#include<stdio.h>
#include<math.h>

void main()
{
int i,n,degree;
float x, sum=0,term;
printf("Enter the value of degree\n");
scanf("%d",&degree);
printf("Enter the value of n\n");
scanf("%d",&n);
x = degree * (3.1416/180);
term = x;
sum=term;
for (i=3;i<=n;i+=2)
{
term=-term*x*x/((i-1)*i);
sum=sum+term;
}
printf("\nThe sine of %d is %f\n", degree, sum);
printf("\nThe sine function of %d using library function is %f\n\n", degree, sin(x));

}
```

10. Write functions to implement string operations such as compare, concatenate, string length. Convince the parameter passing techniques.





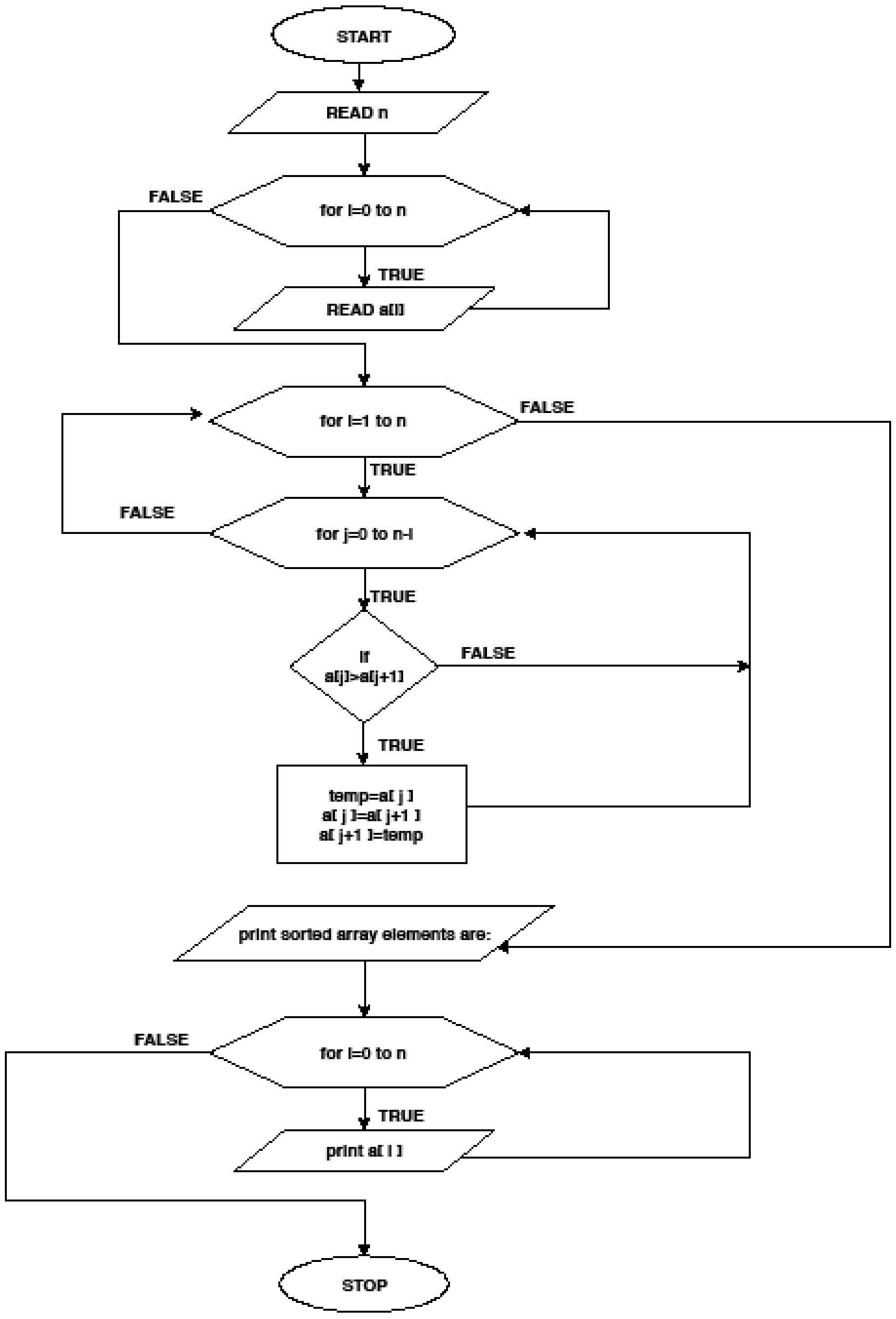



sconcat(str1,str2)
for i=0 to slength(srt1)
FALSE
TRUE
conc[i]=str1[i]
for j=0 to slength(srt2)
FALSE
TRUE
conc[i++]=str1[j]
conc[i]='\0'
print concatenate string: conc
STOP

```
#include<stdio.h>
#include<stdlib.h>
int slength(char str[])
{
   int len=0;
   while(str[len]!='\0')
   len++;
   return len;
}
void scompare(char str1[], char str2[])
{
   int i, flag=0;
   if(slength(str1)!=slength(str2))
   flag=1;
   else
      for(i=0;i<slength(str1);i++)
      {
         if(str1[i]!=str2[i])
            {
               flag=1;
               break;
            }
      }
      if(flag==1)
      printf("\n Strings are not equal\n");
      else
      printf("\n Strings are equal\n");
}
void sconcat(char str1[], char str2[])
{
   char conc[50];
   int i,j;
   for(i=0;i<slength(str1);i++)
      conc[i]=str1[i];
   for(j=0;j<slength(str2);j++)
      conc[i++]=str2[j];

   conc[i]='\0';
   printf("\nConcatenated sting = %s\n",conc);
}
void main()
{
   char s1[50],s2[50];
   int len, choice;
   while(1)
   {
      printf("\n\t MAIN MENU\n");
      printf("\n1. String length\n2. Compare String\n3. Concatenate String\n4. Exit\n");
      printf("\nEnter your Choice.....\n");
      scanf("%d",&choice);
```

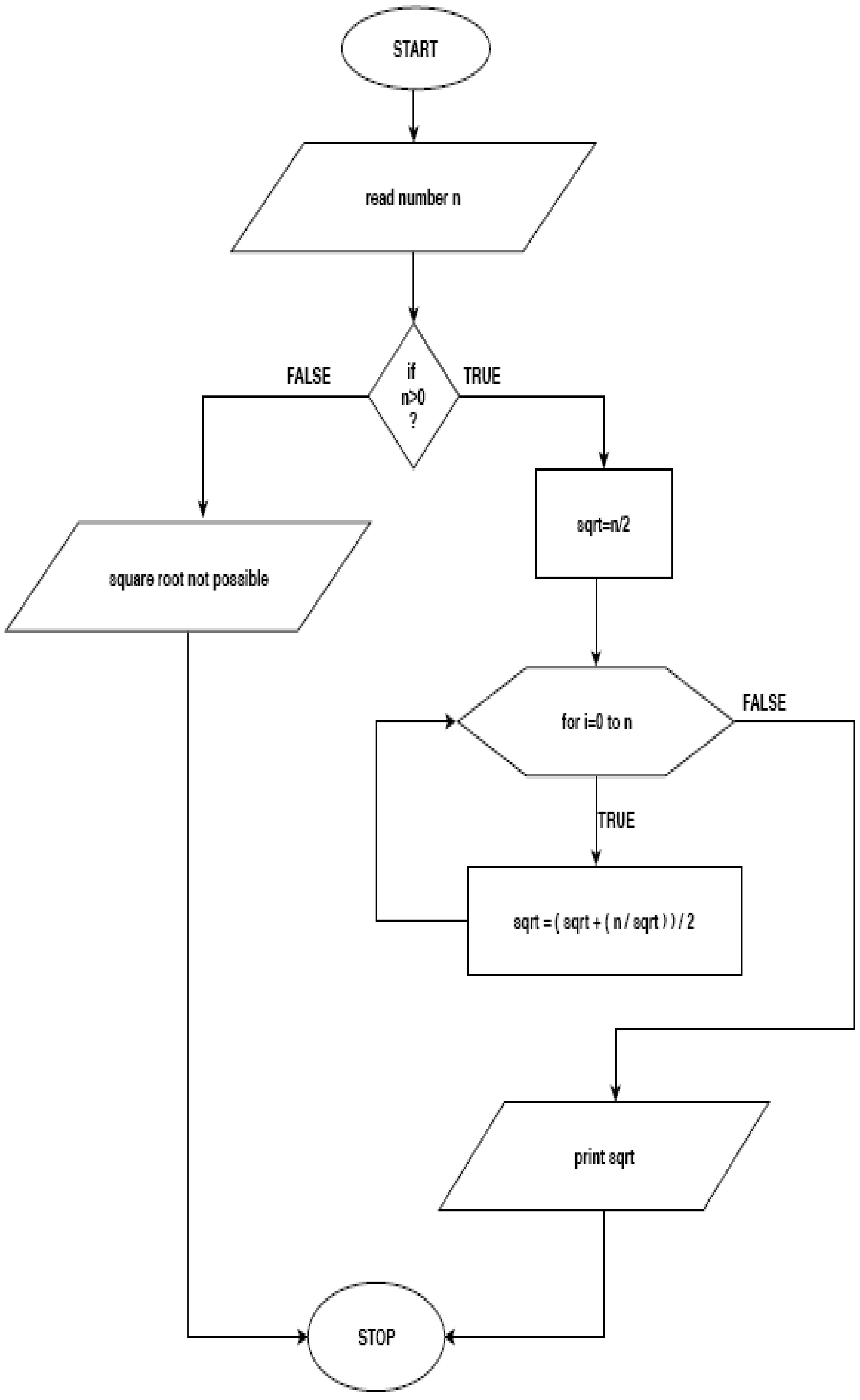
```
          switch(choice)
          {
             case 1: printf("\n Enter the String\n\n");
                   scanf("%s",s1);
                   len=slength(s1);
                   printf("\nThe length of string is %d",len);
                   break;
             case 2: printf("\nEnter the First string\n");
                   scanf("%s",s1);
                   printf("\nEnter the Second string\n\n");
                   scanf("%s",s2);
                   scompare(s1,s2);
                   break;
             case 3: printf("\nEnter the First string\n");
                   scanf("%s",s1);
                   printf("\nEnter the Second string\n");
                   scanf("%s",s2);
                   sconcat(s1,s2);
                   break;
             case 4: exit(0);
             default: printf("\nInvalid choice\n");
          }
       }
    }
```

11. Develop a program to sort the given set of N numbers using Bubble sort.



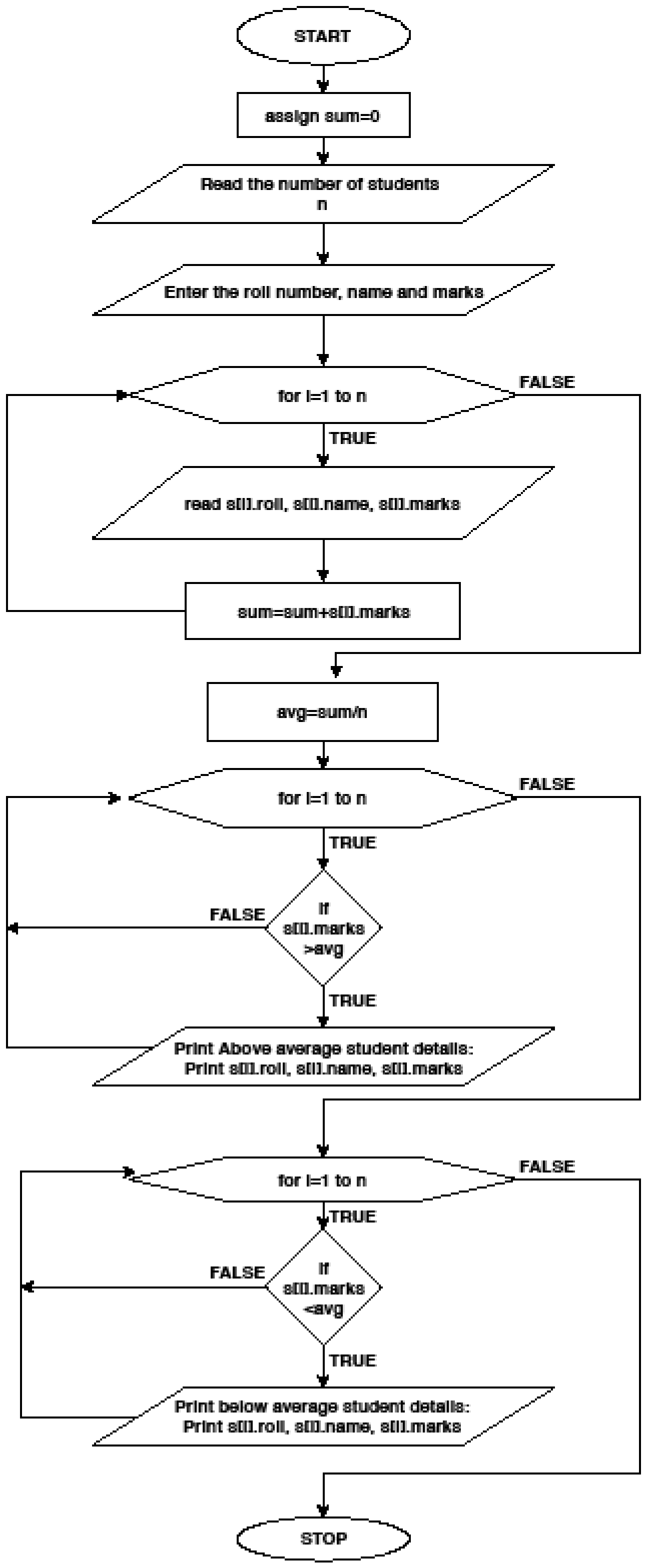
```
#include<stdio.h>
void main()
{
    int i,j,n,temp,a[100];
    printf("\nEnter the size of an array : ");
    scanf("%d",&n);
    printf("\nEnter the elements of an array : \n");
    for(i=0;i<n;i++)
            scanf("%d",&a[i]);
    for(i=1;i<n;i++)
    {
            for(j=0;j<n-i;j++)
            {
                    if(a[j]>a[j+1])
                    {
                            temp=a[j];
                            a[j]=a[j+1];
                            a[j+1]=temp;
                    }
            }
    }
    printf("\nThe elements of the array after sorting : ");
    for(i=0;i<n;i++)
    printf("\n%d",a[i]);

}
```

12. Develop a program to find the square root of a given number N and execute for all possible inputs with appropriate messages. Note: Don't use library function sqrt(n).



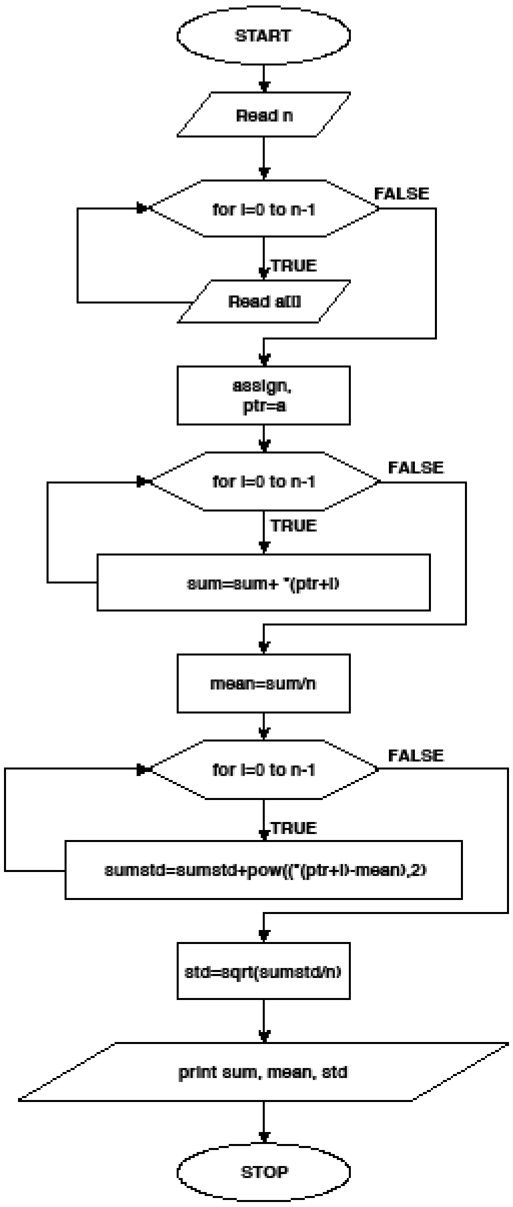
```
#include<stdio.h>

void main( )
{
 float n,sqrt;
 int i;

 printf("\n Enter number:");
 scanf("%f",&n);
 if(n>0)
    {
       sqrt = n/ 2;
       for(i = 0; i <n; i++)
           sqrt = (sqrt + (n/sqrt))/2;
       printf("Square root of %f is %f\n",n,sqrt);
    }
  else
      printf("\n Not possible to find the square root");

}
```

13. Implement structures to read, write, and compute average- marks and the students scoring above and below the average marks for a class of N students.


START
assign sum=0
Read the number of students n
Enter the roll number, name and marks
for i=1 to n
FALSE
TRUE
read s[i].roll, s[i].name, s[i].marks
sum=sum+s[i].marks
avg=sum/n
for i=1 to n
FALSE
TRUE
if s[i].marks >avg
FALSE
TRUE
Print Above average student details: Print s[i].roll, s[i].name, s[i].marks
for i=1 to n
FALSE
TRUE
if s[i].marks <avg
FALSE
TRUE
Print below average student details: Print s[i].roll, s[i].name, s[i].marks
STOP

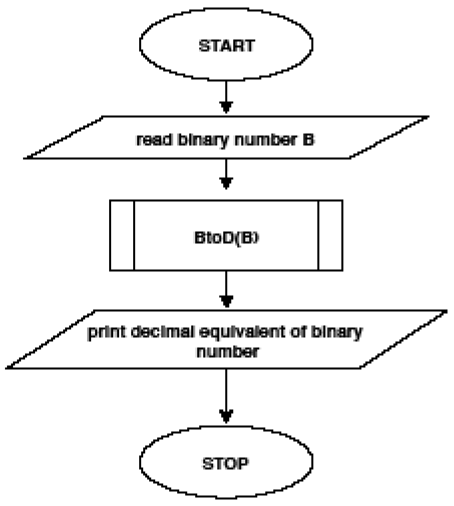
```
#include<stdio.h>
typedef struct
{
   int roll;
   char name[50];
   float marks;
}student;

void main()
{
   student s[50];
   int n,i;
   float sum=0, avg;
   printf("\nEnter the number of students\n");
   scanf("%d",&n);
   printf("\nEnter the roll number, name and marks of\n");
   for(i=1;i<=n;i++)
   {
      printf("\nStudent %d details\n",i);
      scanf("%d%s%f",&s[i].roll,&s[i].name,&s[i].marks);
      sum=sum+s[i].marks;
   }
   avg=sum/n;
   printf("\nAverage marks = %f\n",avg);

   printf("\nStudents details who have scored above average are:\n");
   printf("-----------------------------------------------------\n");
   printf("\nRoll_Number\tName\t\t\tmarks\n");
   for(i=1;i<=n;i++)
   {
      if(s[i].marks>avg)
      printf("\n%d\t\t%s\t\t\t%f\n",s[i].roll,s[i].name,s[i].marks);
   }

   printf("\n\nStudents details who have scored below average are:\n");
   printf("-----------------------------------------------------\n");
   printf("\nRoll_Number\tName\t\t\tmarks\n");
   for(i=1;i<=n;i++)
   {
      if(s[i].marks<avg)
      printf("\n%d\t\t%s\t\t\t%f",s[i].roll,s[i].name,s[i].marks);
   }

}
```

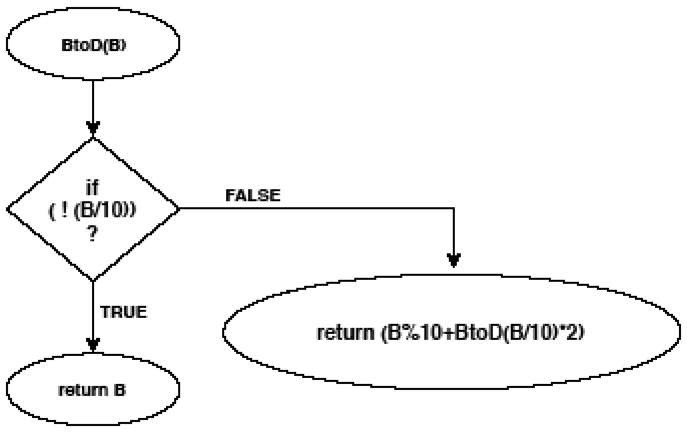
14. Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.



```
#include<stdio.h>
#include<math.h>
void main()
{
   float a[10],*ptr,mean,std,sum=0,sumstd=0;
   int i,n;
   printf("\nEnter the number of elements\n");
   scanf("%d",&n);
   printf("\nEnter the array elements\n");
   for(i=0;i<n;i++)
   scanf("%f",&a[i]);
   ptr=a;
   for(i=0;i<n;i++)
   {
      sum=sum+*(ptr+i);
   }
   mean=sum/n;

   for(i=0;i<n;i++)
   {
      sumstd=sumstd+pow((*(ptr+i)-mean),2);

   }
   std=sqrt(sumstd/n);
   printf("\nSum=%f\n",sum);
   printf("\nmean=%f\n",mean);
   printf("\nStandard deviation=%f\n",std);
}
```

15. Implement Recursive functions for Binary to Decimal Conversion.





```
#include <stdio.h>

int BtoD(int B)
{
   if (!(B / 10))
        return (B);
   return (B % 10 + BtoD(B / 10) * 2);
}

void main()
{
   int B;
   printf("Enter a binary number: ");
   scanf("%d", &B);
   printf("Equivalent decimal number is: %d", BtoD(B));

}
```

VIVA QUESTIONS

1) What is an algorithm?

2) What is high level language?

3) What is compiler?

4) What are tokens?

5) What are identifiers?

6) What are keywords? How many keywords is their in C programming language?

7) What is a variable?

8) What are the rules to be followed while declaring a variable?

9) What is a constant?

10) What is a data type? What are the different data types?

11) What are escape sequence characters?

12) List the size and range of basic data types.

13) What is the difference between a character and string?

14) What is implicit type conversion and explicit type conversion (type casting)?

15) What is precedence of an operator means?

16) What is the difference between printf() and puts() functions.

17) What is function? What are the advantages of functions?

18) What are the different types of functions?

19) What is a library function?

20) What is calling function and called function?

21) What is the meaning of actual parameter and formal parameter?

22) What is the purpose of switch statement? Explain with syntax.

23) What is loop? List the differences between pre-test and post-test loop.

24) What is the meaning of event controlled loop and counter controlled loop?

25) What are the advantages of loops?

26) What is control statement? What are the various types of control statements available in C language?

27) Explain for loop with syntax.

28) What is the difference between while and do-while loop?

29) What are unconditional control statements?

30) What is the use of break statement?

31) What is an array? What is the difference between an ordinary variable and an array variable?

32) What are the differences between recursion and iteration?

33) What is a pointer?

34) What is a NULL pointer?

35) What is a Structure? What are the differences between structures and arrays?

Viva questions with Answer:

1) What is C language?
Ans:
C is a programming language developed at AT & T's Bell Laboratories of USA in 1972. The C programming language is a standardized programming language developed in the early 1970s by Ken Thompson and Dennis Ritchie for use on the UNIX operating system. It has since spread to many other operating systems, and is one of the most widely used programming languages.

2) What is an algorithm?
Ans: An algorithm is a step-by-step method of performing any task.

3) What is a flow chart?
Ans: A flowchart is a type of diagram that represents an algorithm or process, showing the steps as boxes of various kinds, and their order by connecting these with arrows.

4) What is a C Pre-processor?
Ans: C Pre-processor is a program that processes our source program before it is passed to the compiler.

5) What is the use of header files as used in C programming?
Ans: Header files are used to have declarations. It is simple to include a single header file than writing all the needed functions prototypes.

6) What is the Structure of a C Program?
Ans: Documentation, Section, Linking Section, Definition Section, Global declaration Section, main function, subprogram section

7) What is the use of main() function?
Ans: main() is the starting point of program execution.

8) What are the types of constants in c?
Ans: C constants can ba divided into two categories:
Primary constants (Numerical)
Secondary constants (Character)

9) What is a Compiler?
Ans: A **compiler** is a computer program (or set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language, often having a binary form known as object code).

10) What is a Translator?
Ans: A **translator** is a computer program that translates a program written in a given programming language into a functionally equivalent program in a different language.

11) What is a Interpreter?
Ans: An Interpreter is a computer program that directly executes, i.e. *performs*, instructions written in a programming or scripting language, without previously batch-compiling them into machine language.

12) What is a Token in C?
Ans: A Token is the basic building block of a **C. (or)** the basic element recognized by the compiler is the "**token**."
C Tokens are:
Key Words, Identifier, Constants, String – literal, Operator, Punctuators

13) What are Printf() and scanf() Functions :
Ans: printf() and scanf() functions are inbuilt library functions in C which are available in which are available in " stdio.h" header file.
printf() function is used to print the "character, string, float, integer, octal and hexadecimal values" onto the output screen.
scanf() function is used to read character, string, numeric data from keyboard.

14) What is a Data Type and List the different Data types?
Ans: C data types are defined as the data storage format that a variable can store a data to perform a specific operation.
List of Data Types:

1) **Basic Data Types:** Int, Float, Char, Double, long int
2) **Enumeration Data Type:** enum
3) **Derived Data Type:** Pointer, array
4) **User Defined Type:** structure, union
5) **Void Data Type:** void

15) What is enum Data Type?
Ans:
· Enumeration data type consists of named integer constants as a list.
· It start with 0 (zero) by default and value is incremented by 1 for the sequential identifiers in the list.

Syntax:
enum identifier { enumerator-list };

16) What is a Void Data Type?
Ans: Void is an empty data type that has no value.

17) What is a comment in C?
Ans: Comments are like helping text in your C program and they are ignored by the compiler.
We can write in between /* and */ or // (Line Comments)

18) What is an Identifier in C?
Ans: A C identifier is a name used to identify a variable, function, or any other user-defined item. An identifier starts with a letter A to Z or a to z or an underscore _ followed by zero or more letters, underscores, and digits (0 to 9). C does not allow punctuation characters such as @, $, and % within identifiers. C is a **case sensitive** programming language.

19) What is a Key word in C?
Ans: Keywords are reserved words in C and Keywords are may not be used as constant or variable or any other identifier names.

20) How many Keywords are there in C and List out the Keywords?
Ans : There are 33 reserved keywords are there in C. They are:

auto	Else	long	switch
break	Enum	register	typedef
case	Extern	return	union
char	Float	short	unsigned
const	For	signed	void
continue	Goto	sizeof	volatile
default	If	static	while
do	Int	struct	_Packed
double			

21) Define a Variable?
Ans: A variable is nothing but a name given to a storage area that our programs can manipulate. Each variable in C has a specific type, which determines the size and layout of the variable's memory.
Syntax for variable Declaration: DataType VariableList;
int i,j; cahr c, ch;

22) What are the steps to develop a C Program?
· Specifying the problem statement
· Designing an algorithm
· Coding
· Debugging
· Testing and Validating
· Documentation and Maintenance.

23) List some Syntactic Errors or Syntax Errors?
Ans:
1) Missing semicolon(Statement Missing ;)
2) Undeclared a variable name or Undefined symbol (Then check variable declaration syntax and check for header file for some keywords)
3) ’)’ expected (Then check the no of parenthesis o pened and closed)
4) Illegal string constant (check for the last double quote in a string)
5) printf and scanf arguments should be placed in ( )
6) Compound statement missing (Check for the no of { and } are opened and closed)
7) Proto type missing error (Check for Function declaration statement)
8) Forgetting to put &,”, and comma operator in spe cific places.
9) Comparing strings with == operator

24) List some Warnings?
Ans:
1) Miss use of = and ==
2) Loop has no body (remove the semicolon at last of the for loop or while loop)
3) Uninitialized a variable

25) What is type casting?
Ans: Converting a variable of one type to another type.

26) What are Iterative or Looping Statements?
Ans: Iterative or Looping statement which executes the statements with in the compound statement by checking the condition, and performs same set of statements as a iterative process or as a loop until the condition false.
The Iterative or Looping Statements are: While, do-while, for

27) What is the difference between for loop and while loop?
Ans: For Loop is used to execute a set of statements in fixed number of times. We use While loop when the number of iterations to be performed is not known in advance we use while loop.

28) What are Unconditional Statements?
Ans: goto and labeled statements, break Statement continue Statement.

29) Define Break Statement, Write syntax for Break statement?
Ans: The **break** statement terminates the execution of the nearest enclosing **do**, **for**, **switch**, or **while** statement in which it appears. Control passes to the statement that follows the terminated statement.
Syntax:
break;

30) Define Continue Statement, Write syntax for Continue statement?
Ans: The **continue** statement passes control to the next iteration of the nearest enclosing **do**, **for**, or **while** statement in which it appears
Syntax:
continue;

31) What is a Macro?
Ans: Macros are the identifiers that represent statements or expressions. To associate meaningful identifiers with constants, keywords, and statements or expressions.

32) Define Storage class with Syntax?
Ans: Storage class specifiers in C language tells the compiler where to store a variable, how to store the variable, what is the initial value of the variable and life time of the variable.
Syntax: storage_specifier data_type variable _name

33) What are Different Storage Classes, Explain in brief?
Ans: There are 4 storage class specifiers available in C language. They are,
1) Auto
2) Extern
3) Static
4) Register

34) Define Pointer with Syntax and example?
Ans: C Pointer is a variable that stores/points the address of another variable. C Pointer is used to allocate memory dynamically i.e. at run time.
Syntax: data_type *var_name; **Example :** int *p; char *p;

35) What are the uses of Pointers?
Ans:
- *Pointer is used in the following cases*
- *It is used to access array elements.*
- *It is used for dynamic memory allocation.*
- *It is used in Call by reference.*
- *It is used in data structures like trees, graph, linked list etc.*

36) What is a pointer to pointer?
Ans: If a pointer variable points another pointer value. Such a situation is known as a pointer to a pointer.
***Example :** int *p1, **p2, v=10;P1=&v; p2=&p1;*
Here p2 is a pointer to a pointer.

37) What are the Advantages of Functions? Ans:
- It reduces the Complexity in a program by reducing the code.
- Function are easily understanding and reliability and execution is faster.
- It also reduces the Time to run a program.In other way, Its directly proportional to Complexity.
- Its easy to find-out the errors due to the blocks made as function definition outside the main function.

38) What is Recursion?
Ans: A recursion function is one which calls itself either directly or indirectly it must halt at a definite point to avoid infinite recursion.

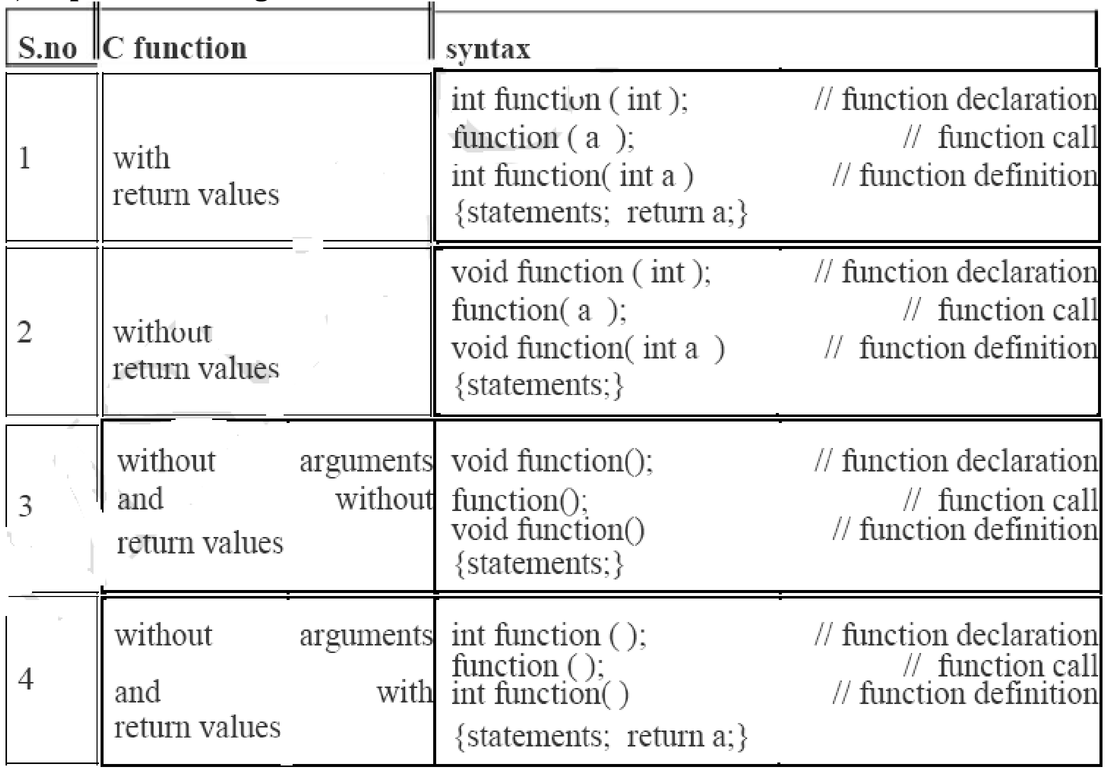
39) Define Function Declaration, Function Call and Function Definition with Syntaxes:
Ans:
Function declaration or prototype - This informs compiler about the function name, function parameters and return value's data type.
Function call – This calls the actual function
Function definition – This contains all the stateme nts to be executed.

S.no	**C function aspects**		**syntax**
1	function definition		return_type function_name (arguments list) { Body of function; }
2	function call		function_name (arguments list);
3	Function declaration		return_type function_name (argument list);

40) Define Call by Value?
Ans: In call by value method, the value of the variable is passed to the function as parameter. The value of the actual parameter can not be modified by formal parameter. Different Memory is allocated for both actual and formal parameters.
· **Actual parameter** – This is the argument which is us ed in function call.
· **Formal parameter** – This is the argument which is us ed in function definition

41) Define Call by Reference?
Ans: In call by reference method, the address of the variable is passed to the function as parameter. The value of the actual parameter can be modified by formal parameter. Same memory is used for both actual and formal parameters since only address is used by both parameters.

42) Explain the Categories of Functions?



S.no	C function	syntax

1	with return values	int function (int); // function declaration
		function (a); // function call
		int function(int a) // function definition
		{statements; return a;}

2	without return values	void function (int); // function declaration
		function(a); // function call
		void function(int a) // function definition
		{statements;}

3	without arguments and without return values	void function(); // function declaration
		function(); // function call
		void function() // function definition
		{statements;}

4	without arguments and with return values	int function ( ); // function declaration
		function ( ); // function call
		int function( ) // function definition
		{statements; return a;}

43) What is an Argument?
Ans: An argument is an entity used to pass data from the calling to a called function.

44) What are Built-in-Functions/ Pre Defined Functions / Library Functions?
Ans: The functions that are predefined and supplied along with the compiler are known as built in functions. They are also known as library functions.

45) What are the uses of Functions?
Ans:
C functions are used to avoid rewriting same logic/code again and again in a program. There is no limit in calling C functions to make use of same functionality wherever
required.
We can call functions any number of times in a program and from any place in a program.
A large C program can easily be tracked when it is divided into functions.
The core concept of C functions are, re-usability, dividing a big task into small pieces to achieve the functionality and to improve under standability of very large C programs.

Success
is dependent on effort.
- Sophocles
All the Best