TABLE OF CONTENTS

14.		To swap two numbers using call by value and call by reference.			
15.		To find the sum of an integer array using pointers.			
16.		To find the maximum element in an integer array using pointers.			
17.		To generate salary slip of employees using structures and pointers.			
18.		To display the contents of the file on the monitor screen.			
19.		Getting the input from the keyboard and retrieve the contents of the file using file operation commands			
20.		To create two files with a set of values. Merge the two file contents to form a single file			
21.		To pass the parameter using commands line arguments.			
22.		To find the factorial of a given number using Functions in C.			
23.		To check whether the given string is Palindrome or not.			
24.		To check whether the given value is Prime or not.			
25.		To Replace all 0's with 1's in a Number.			

Ex. No : 01 TO FIND THE AREA OF THE TRIANGLE

DATE :

AIM:

To find the area of the triangle.

ALGORITHM:

Step 1: Start the program

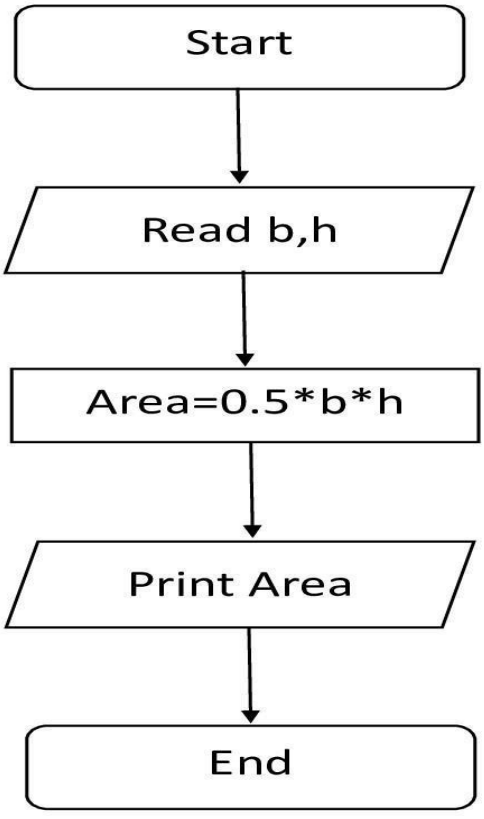
Step 2: Input the breath and height of the triangle.

Step 3: Calculate the area of the triangle using the formula area = (b*h) / 2.

Step 5: Print the Area.

Step 6: End the program.

FLOW CHART:

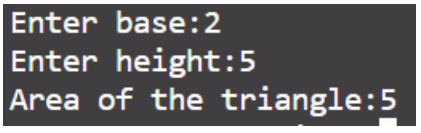


PROGRAM:

/*To find the area of the triangle*/

```
#include <stdio.h>
int main()
 {
float base, height,
int area;
printf("Enter base:");
scanf("%f", &base);
printf("Enter height:");
scanf("%f", &height);
area = (base * height) / 2;
printf("Area of the triangle : %.d\n", area);
return 0;
}
```

OUTPUT:



Criteria	Maximum marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus the c program to implement finding the area of the triangle has been executed and the output is verified.

Ex. No: 02 TOTAL AND AVERAGE PERCENTAGE OBTAINED BY STUDENTS

DATE:

AIM:

To find the total, average percentage of a student.

ALGORITHM:

Step 1: Start the program

Step 2: Input marks of five subjects.

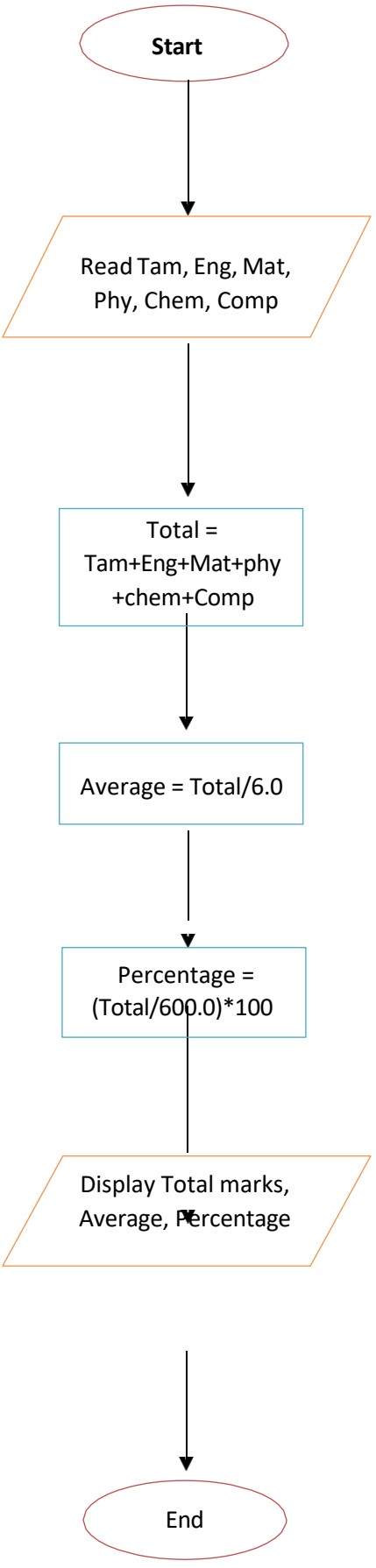
Step 3: Calculate sum of all subjects and store in total = eng + phy+ chem + math +comp

Step 4: Divide sum of all subjects by total number of subject to find average i.e. average = total / 5

Step 5: Print the sum and average result

Step 6: End the program.

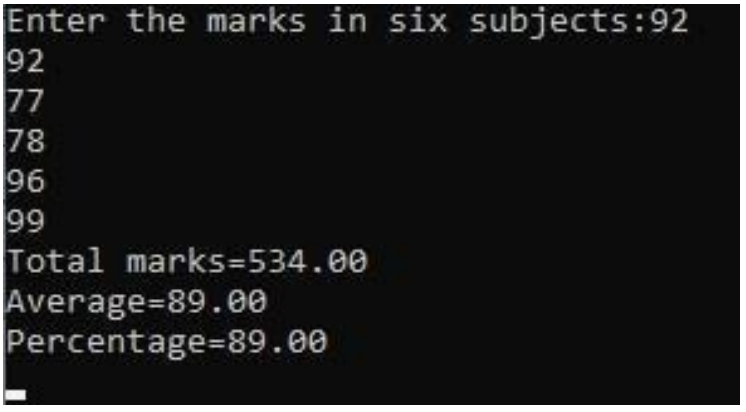
FLOW CHART:


Start
Read Tam, Eng, Mat, Phy, Chem, Comp
Total = Tam+Eng+Mat+phy +chem+Comp
Average = Total/6.0
Percentage = (Total/600.0)*100
Display Total marks, Average, Percentage
End

PROGRAM:

```
/*To find the total, average percentage of a student*/
#include<stdio.h>
#include<conio.h>
void main()
{
    //declaration of variables
    float tam,eng,mat,phy,chem,comp;
    float total,average,percentage;
    printf("Enter the marks in six subjects:");
    //getting input for six subjects
    scanf("%f%f%f%f%f%f",&tam,&eng,&mat,&phy,&chem,&comp);
    //adding all the six subjects to get total
    total=tam+eng+mat+phy+chem+comp;
    //dividing the total value by total no of subjects
    average=total/6.0;
    percentage=(total/600.0)*100;
   //printing the output values
   printf("Total marks=%.2f\n",total);
   printf("Average=%.2f\n",average);
   printf("Percentage=%.2f\n",percentage);
   getch();
}
```

OUTPUT:



Criteria	Maximum marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to implement finding the total and average marks has been executed and the output is verified.

Ex. No: 03 TO READ 3 DIGIT NUMBER AND PRINT OUTPUT AS 1 HUNDREDS 7 TENS 3 UNITS FOR 172 AND PRINT THE REVERSE OF THE NUMBER.

DATE:

AIM:

To read a 3 digit number and output as 1 hundreds 7 tens two units for 172 and print the reverse of the number.

ALGORITHM:

STEP 1: Start the program

STEP 2: Declare the variables of integer datatype as no, hun, ten, unit, rev

STEP 3: Initialize the variable r=0

STEP 4: Get the number you want to reverse and find the ones hundreds tens and units

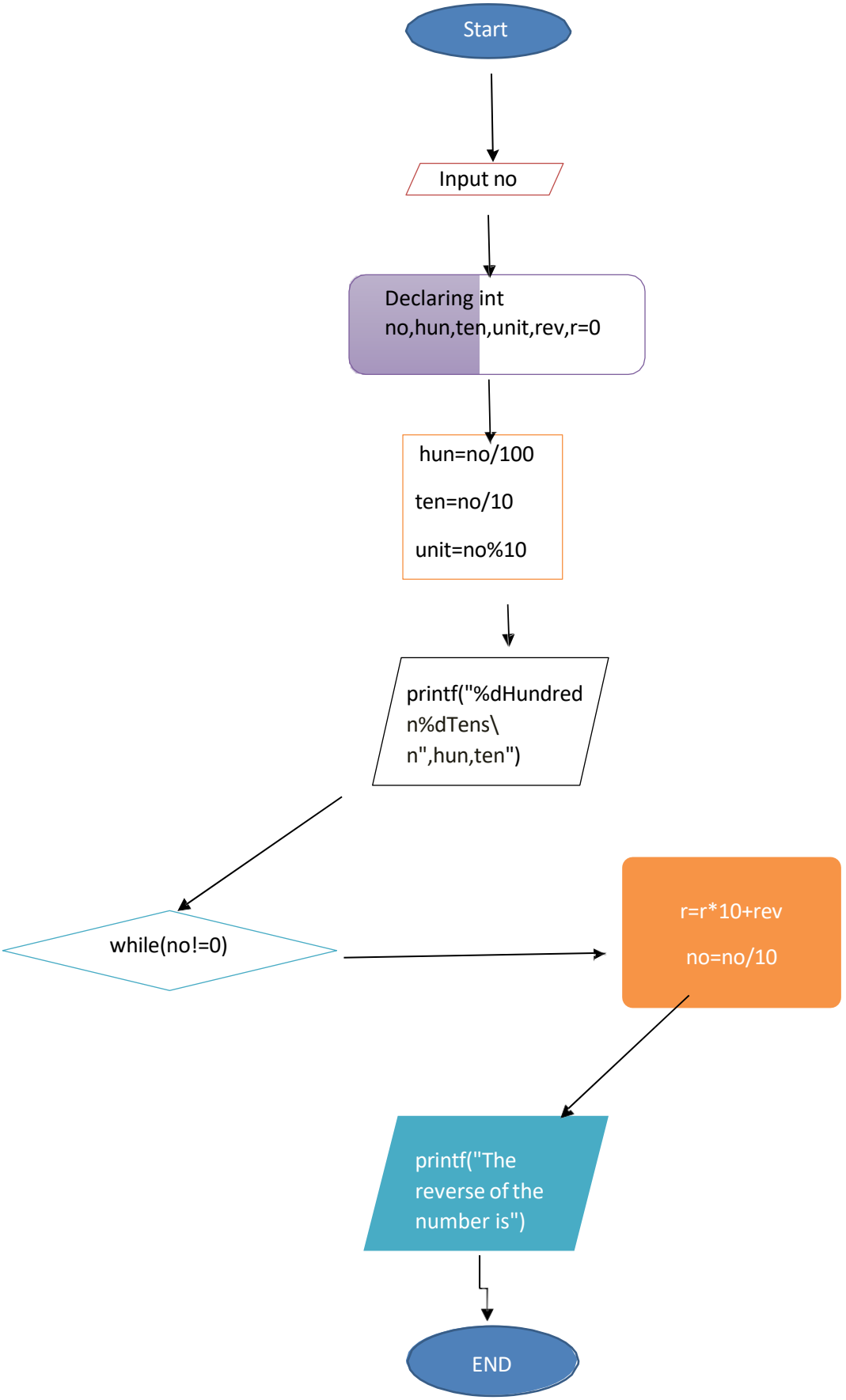
STEP 5: hun =no/100 **STEP 6:** ten = no/10 **STEP 7:** unit = no/10

STEP 8: And using while loop find the reverse of the number

STEP 9: Run the code and get the output

STEP 10: End the program

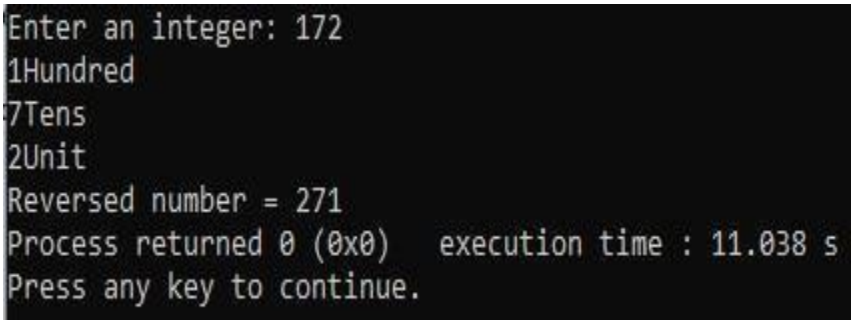
FLOW CHART:


Start
Input no
Declaring int no,hun,ten,unit,rev,r=0
hun=no/100
ten=no/10
unit=no%10
printf("%dHundred n%dTens\ n",hun,ten")
while(no!=0)
r=r*10+rev
no=no/10
printf("The reverse of the number is")
END

PROGRAM:

```
#include<stdio.h>
int main()
 {
   int no, reverse = 0, remainder,hun,ten,unit;
   printf("Enter an integer: ");
   scanf("%d", &no);
   hun=no/100;
   ten=((no/10)%10);
   unit=no%10;
   printf("%dHundred\n%dTens\n%dUnit\n",hun,ten,unit);
    //using while loop getting the reverse of number
   while (no != 0)
    {
        remainder = no % 10;
        reverse = reverse * 10 + remainder;
        no /= 10;
    }
      printf("Reversed number = %d", reverse);
      return 0;
}
```

OUTPUT:



Criteria	Maximum marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

This program is to implement the ones tens hundreds and units of a given number 172 and the reverse of the same number the code was run and the output is verified.

EX. NO:04 CHECK WHETHER THE GIVEN CHARACTER IS VOWEL OR NOT USING SWITCH-CASE

DATE:

AIM:

To check whether a given character is vowel or not using Switch – Case.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare the variables to store the character.

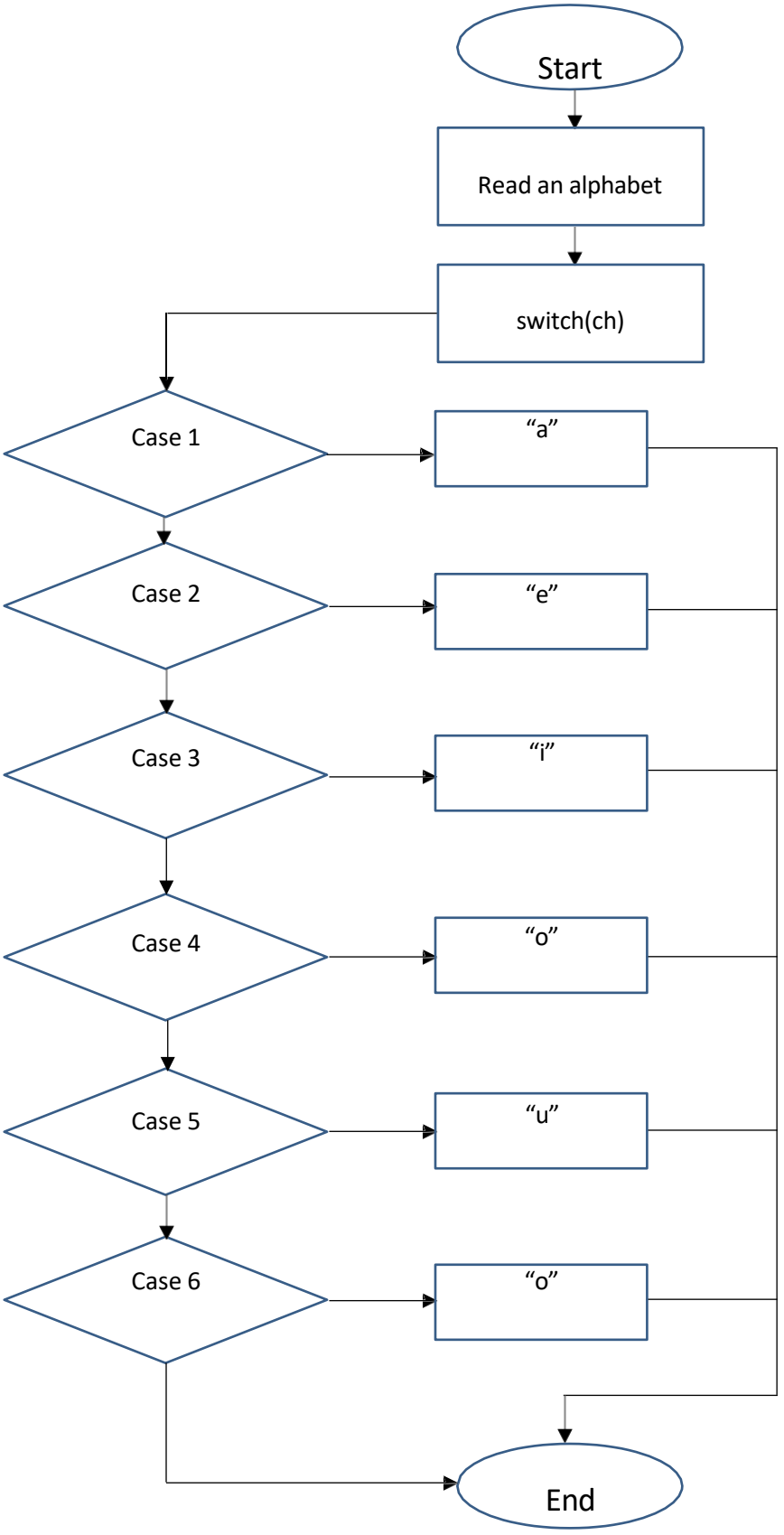
Step 3: Get an character.

Step4: Use switch cases to find the vowels and consonants.

Step 5: Print them as output.

Step 6: Stop the program.

FLOWCHART:

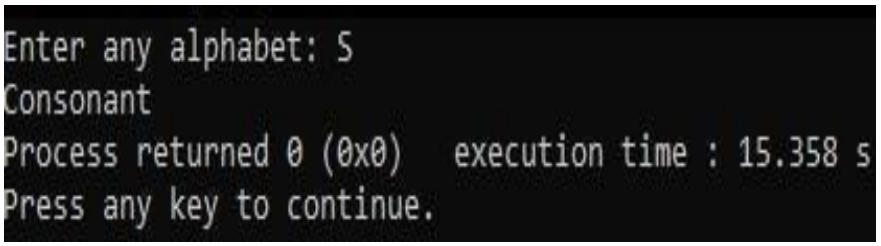

Start
Read an alphabet
switch(ch)
Case 1
"a"
Case 2
"e"
Case 3
"i"
Case 4
"o"
Case 5
"u"
Case 6
"o"
End

PROGRAM:

/*To check whether a given character is vowel or not using Switch – Case*/

```
#include <stdio.h>
int main()
{
  char ch;
  printf("Enter any alphabet: ");
  scanf("%c", &ch);
  switch(ch)
  {
    case 'a':
      printf("Vowel");
      break;
   case 'e':
      printf("Vowel");
      break;
   case 'i':
      printf("Vowel");
      break;
   case 'o':
     printf("Vowel");
     break;
   case 'u':
      printf("Vowel");
      break;
   case 'A':
     printf("Vowel"); break;
     case 'E':
     printf("Vowel");
     break;
  case 'I':
     printf("Vowel");
     break;
   case 'O':
       printf("Vowel");
       break;
   case 'U':
      printf("Vowel");
      break;
   default:
      printf("Consonant");
 }
 return 0;
}
```

OUTPUT:



Criteria	Maximum marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to find whether the given character is vowel or not has been executed and the output is verified.

EX. NO:05 PRINT THE NUMBERS FROM 1 TO 10 ALONG WITH THEIR SQUARE

DATE:

AIM:

To print the numbers from 1 to 10 along with their squares.

ALGORITHM:

STEP 1: Start the program.

STEP 2: Declare the variables to be used .

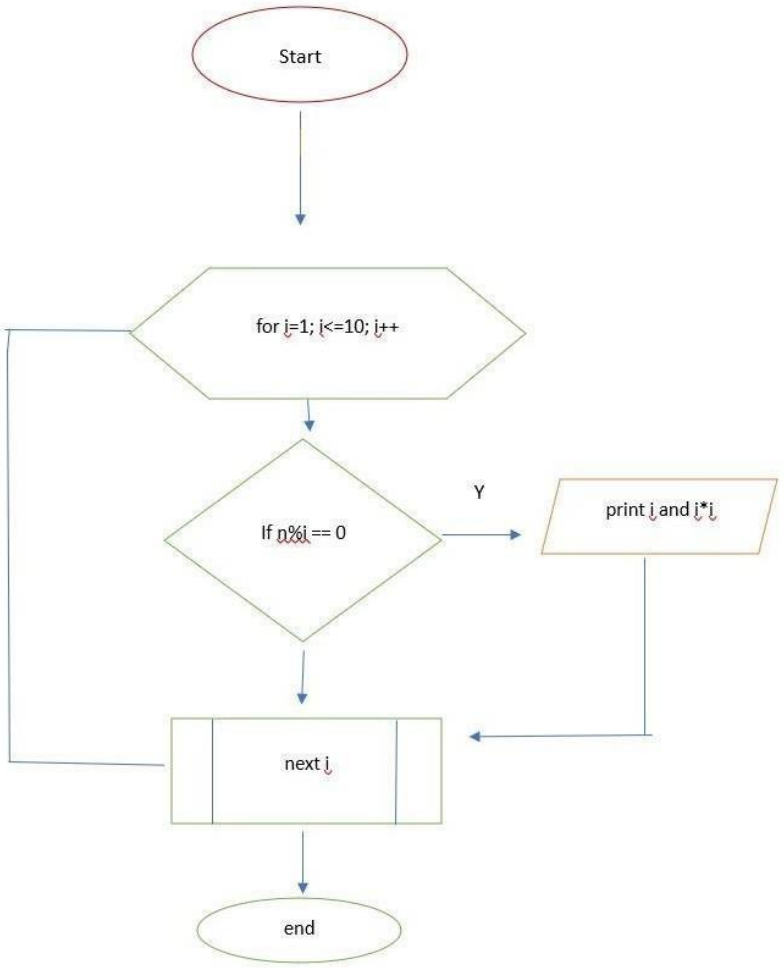
STEP 3: Initialize a loop to iterate from 1 to 10.

STEP 4: Inside the loop, print the current number and its square.

STEP 5: Close the loop.

STEP 6: End the program.

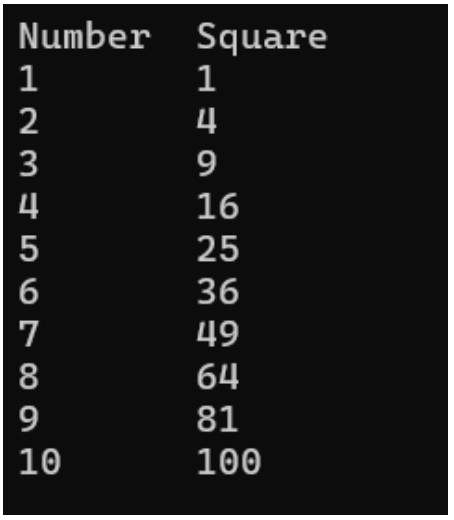
FLOW CHART:



PROGRAM:

```
/* To print the numbers from 1 to 10 along with their squares */

#include <stdio.h>

int main() {
printf("Number\tSquare\n");

// Loop to print numbers and their squares
for (int i = 1; i <= 10; ++i) {
// Print current number and its square
printf("%d\t%d\n", i, i * i);
}

return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to implement printing the numbers from 1 to 10 along with their squares has been executed and the output is verified.

EX .NO:06 **SUM OF N NUMBERS USING FOR,**

WHILE, DO-WHILE LOOP

DATE:

AIM:

To find the sum of "n" numbers using for, while and do – while statements.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare the variables to get the numbers for input.

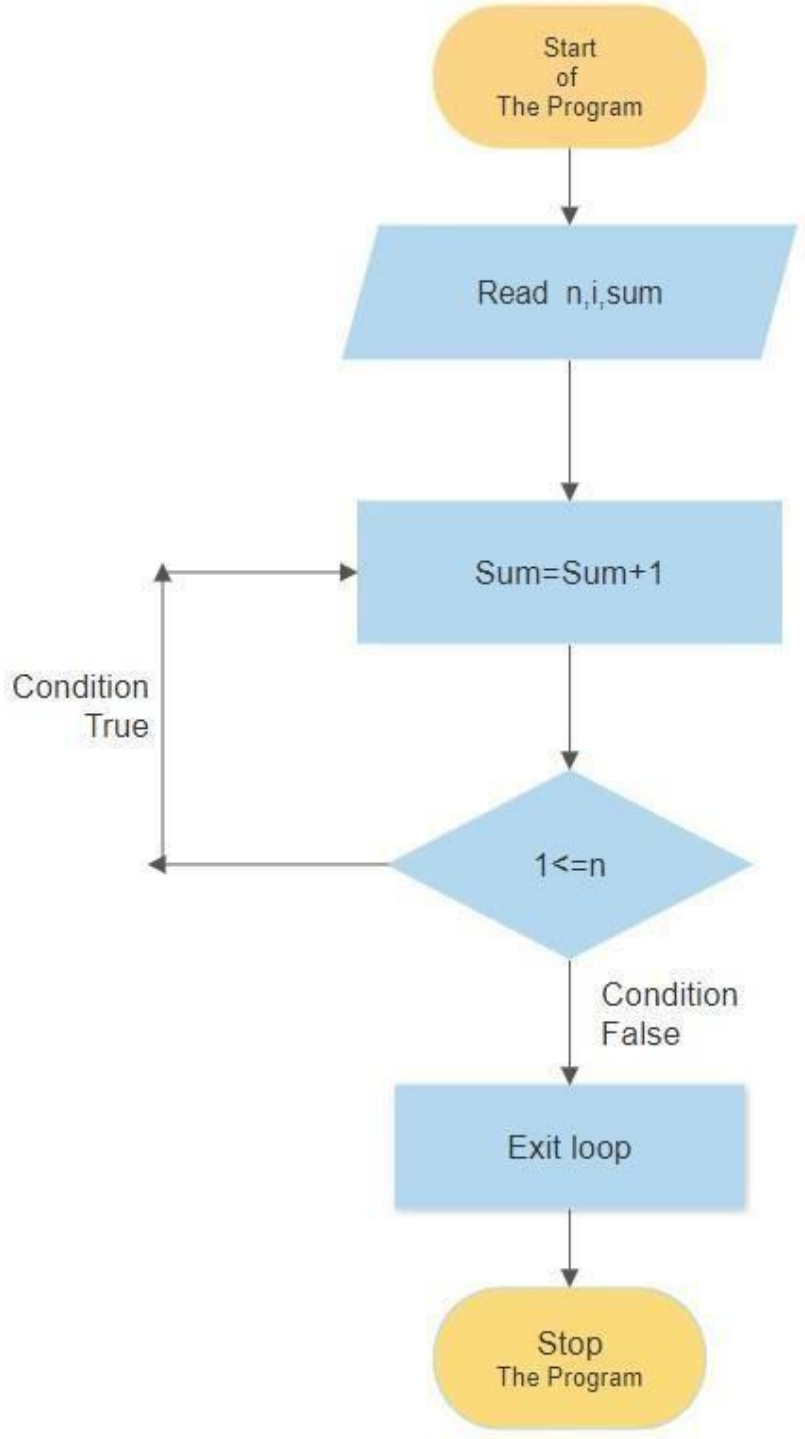
Step 3: Get the total number n.

Step 4: Use the loop concepts to calculate the numbers.

Step 5: Print them as output.

Step 6: Stop the program.

FLOW CHART:

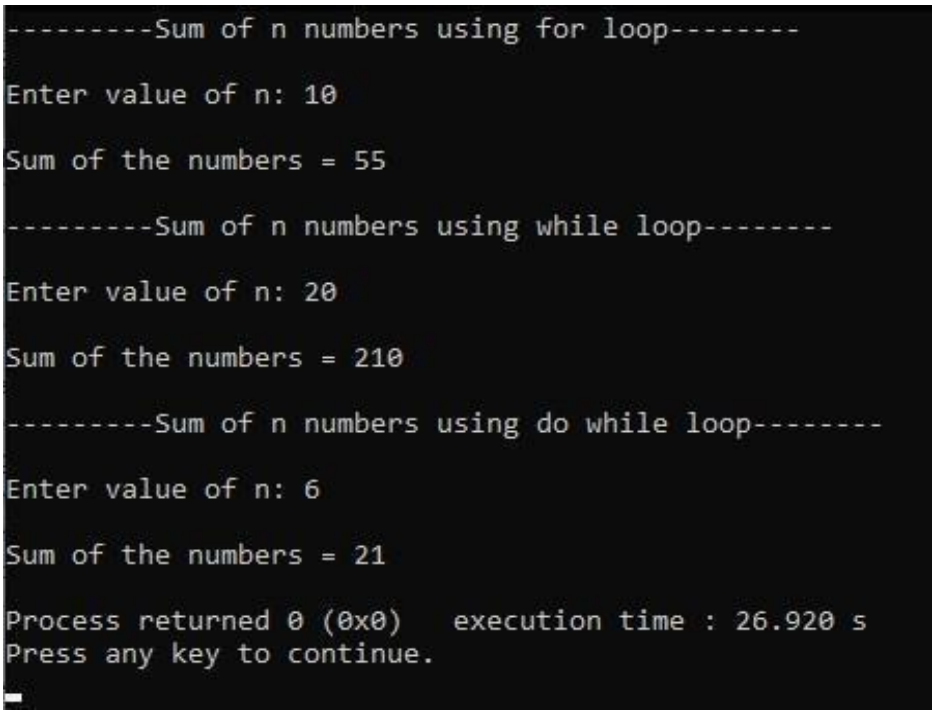

Start
of
The Program
Read n,i,sum
Sum=Sum+1
Condition
True
1<=n
Condition
False
Exit loop
Stop
The Program

PROGRAM:

/* To find the sum of "n" numbers using for, while and do – while statements*/

```
#include <stdio.h>
int main()
{
        int n,n1,n2,i,j=1,k=1,s=0,s1=0,s2=0;
        printf("---------Sum of n numbers using for loop\n");
        printf("\nEnter value of n: ");
        scanf("%d",&n); for(i=1;i<=n;i++)
{
        s+=i;
}
printf("\nSum of the numbers = %d\n",s);
printf("\n---------Sum of n numbers using while loop \n");
printf("\nEnter value of n: ");
scanf("%d",&n1); while(j<=n1)
{
        s1+=j; j++;
}
printf("\nSum of the numbers = %d\n",s1);
printf("\n---------Sum of n numbers using do while loop\n");
printf("\nEnter value of n: "); scanf("%d",&n2);
do
{
        s2+=k; k++;
}
while(k<=n2);
printf("\nSum of the numbers = %d\n",s2); return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to implement printing the numbers from 1 to 10 along with their squares has been executed and the output is verified.

EX .NO: 07 **STRING OPERATION**

DATE:

AIM:

To perform various string handling functions: strlen , strcpy , strcat , strcmp.

ALGORITHM:

Step 1: Start the program

Step 2: Declare the Variables

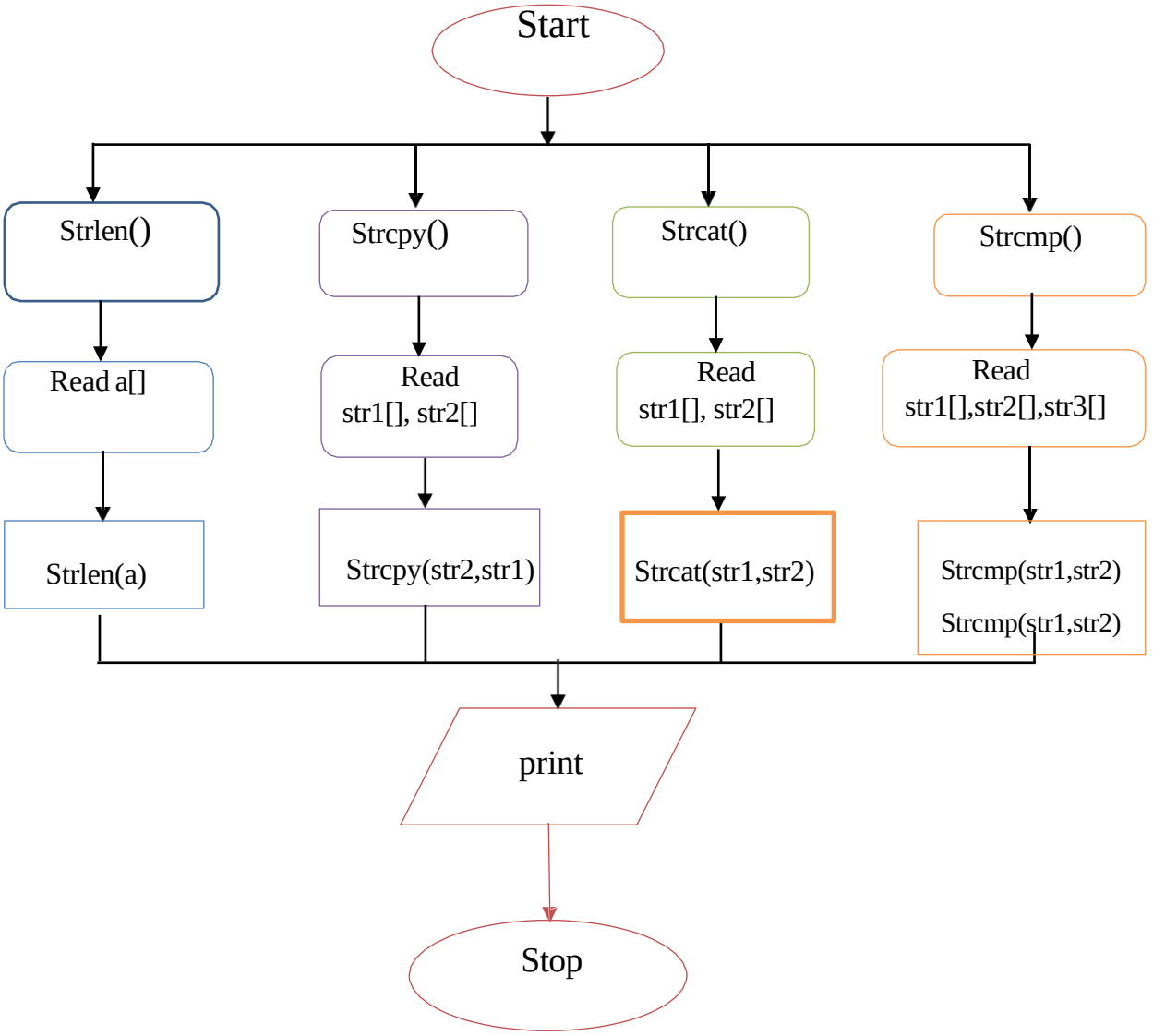
Step 3: Enter the value of 3strings

Step 4: Enter the string functions to perform string handling functions

Step 5: Display the output

Step 6: End the program

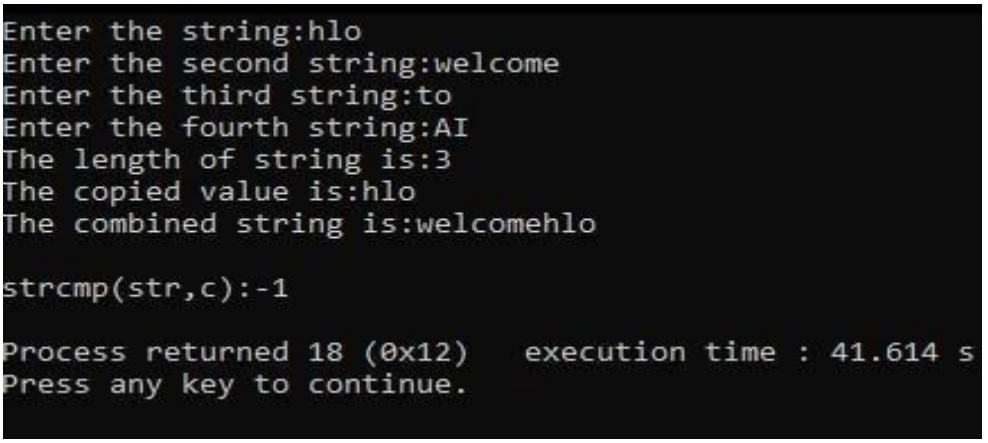
FLOWCHART:


Start
Strlen()
Strcpy()
Strcat()
Strcmp()
Read a[]
Read
str1[], str2[]
Read
str1[], str2[]
Read
str1[],str2[],str3[]
Strlen(a)
Strcpy(str2,str1)
Strcat(str1,str2)
Strcmp(str1,str2)
Strcmp(str1,str2)
print
Stop

PROGRAM:

```
/* To perform various string handling functions: strlen , strcpy , strcat , strcmp*/
#include<stdio.h>
#include<string.h>
void main()
{
   char str[100],c[100],e[100],f[100],t[100];
   int g;
   printf("Enter the string:");
   scanf("%s",&str);
   printf("Enter the second string:");
   scanf("%s",&c);
   printf("Enter the third string:");
   scanf("%s",&e);
   printf("Enter the fourth string:");
   scanf("%s",&f);
   printf("The length of string is:%d\n",strlen(str));
   printf("The copied value is:%s\n",strcpy(t,str));
   printf("The combined string is:%s\n",strcat(c,str));
   printf("\nstrcmp(str,c):%d\n",strcmp(str,c));

}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to perform string function has been executed and the output is verified.

EX .NO: 08 REMOVAL OF CHARACTERS

DATE:

AIM:

To remove all characters in a string except alphabets.

ALGORITHM:

Step 1: Start the program

Step 2: Enter the string

Step 3: Traverse the string, character by character

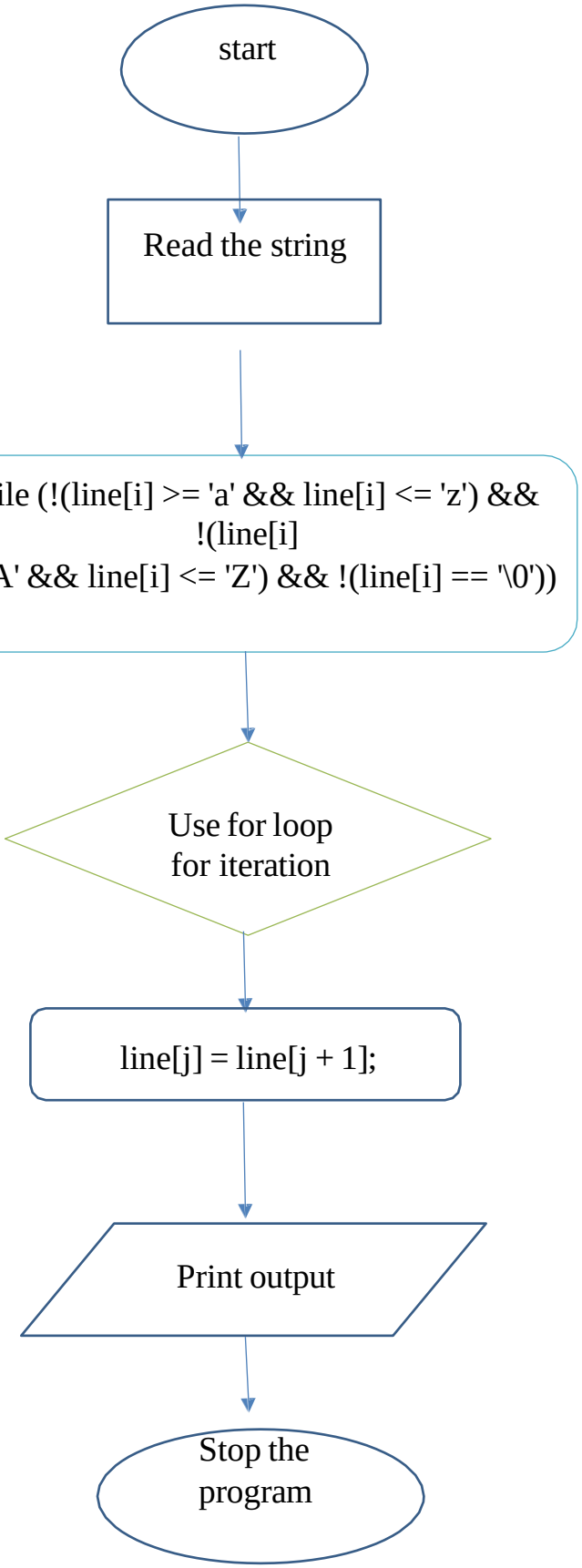
Step 4: If the character is not an alphabet, do not add it to the resultant string

Step 5: If the character is an alphabet, add it to the resultant string

Step 6: Print the output

Step 7: End the program

FLOWCHART:

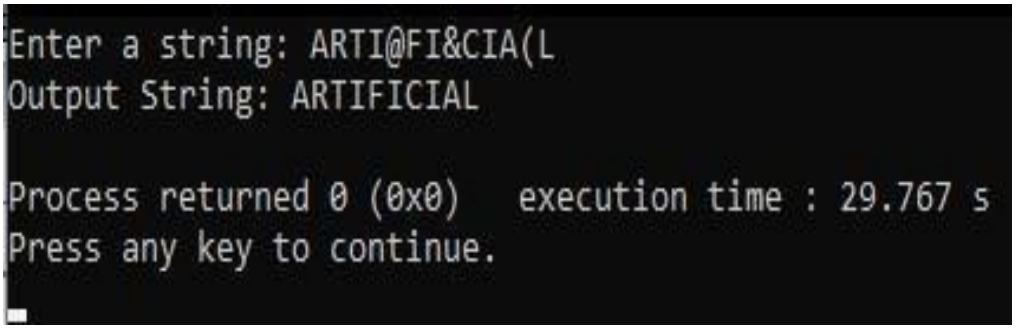

start
Read the string
while (!(line[i] >= 'a' && line[i] <= 'z') && !(line[i]
>= 'A' && line[i] <= 'Z') && !(line[i] == '\0'))
Use for loop
for iteration
line[j] = line[j + 1];
Print output
Stop the
program

PROGRAM:

/* To remove all characters in a string except alphabets*/

```
#include <stdio.h>
int main()
{
  char line[150]; printf("Enter
  a string: ");
  fgets(line, sizeof(line),stdin);
  for (int i = 0, j; line[i] != '\0'; ++i)
  {
    while (!(line[i] >= 'a' && line[i] <= 'z') && !(line[i] >= 'A' && line[i] <= 'Z') &&
!(line[i] == '\0'))
    {
      for (j = i; line[j] != '\0'; ++j)
      {
        line[j] = line[j + 1];
      }
      line[j] = '\0';
    }
  }
  printf("Output String: ");
  puts(line);
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to remove characters from a string has been executed and the output is verified.

EX .NO: 09 SMALLEST AND LARGEST NUMBER

DATE:

AIM:

To find the largest and smallest element in an array.

ALGORITHM:

Step 1: Start the program.

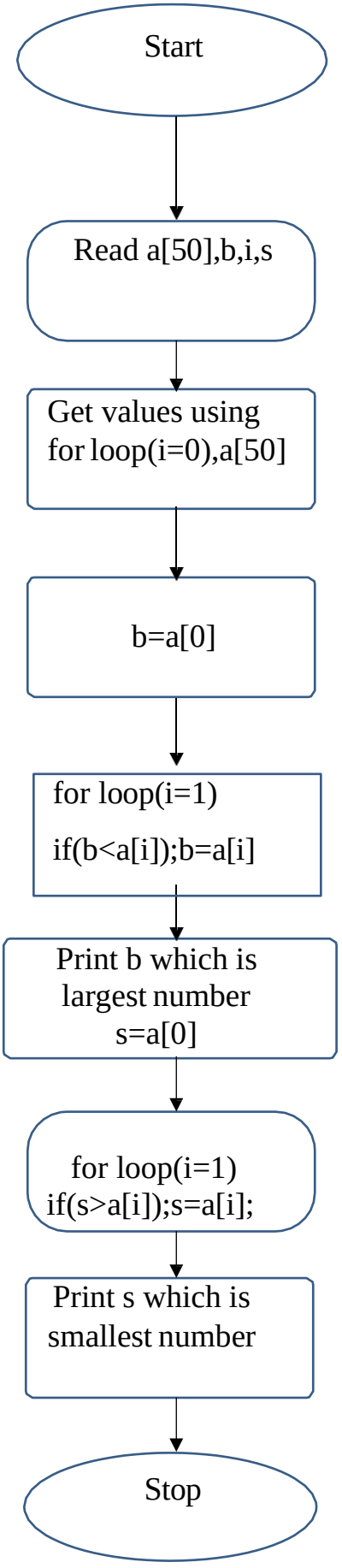
Step 2: Get the array where we have to find the largest and smallest number.

Step 3: Sort the array in Ascending order or descending order.

Step 4: Print the first element which is smallest and last element which is largest if sorted in ascending order and vice versa for descending order.

Step 5: Stop the program.

FLOWCHART:


Start
Read a[50],b,i,s
Get values using for loop(i=0),a[50]
b=a[0]
for loop(i=1)
if(b<a[i]);b=a[i]
Print b which is largest number
s=a[0]
for loop(i=1)
if(s>a[i]);s=a[i];
Print s which is smallest number
Stop

PROGRAM:

/* To find the largest and smallest element in an array*/

```
#include<stdio.h>
int main()
{
   int a[100],num,i,j,temp;
   printf("Enter how many number u want to enter the array\n");
   scanf("%d",&num);
   printf("Enter the numbers \n");
   for(i=0;i<num;i++)
   {
     scanf("%d",&a[i]);
   }
   for(i=0;i<num;i++)
   {
      printf("%d\t",a[i]);
   }
   for(i=0;i<num;i++)
   {
      for(j=i+1;j<num;j++)
          if(a[i]<a[j])
      {
```

```
            temp=a[i];
            a[i]=a[j];
            a[j]=temp;

        }
    }
    printf("\nthe sorted array is:");
      for(i=0;i<num;i++)
      {
         printf("%d\t",a[i]);
      }
      printf("\nthe greatest number is %d",a[0]);
      printf("\n The smallest number is %d",a[num-1]);

}
```

OUTPUT:

```
Enter how many number u want to enter the array
8
Enter the numbers
34
2
90
76
21
56
77
43
34      2       90      76      21      56      77      43
the sorted array is:90  77      76      56      43      34      21      2
the greatest number is 90
 The smallest number is 2
Process returned 0 (0x0)   execution time : 21.733 s
Press any key to continue.
```

Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to find largest and smallest numbers in an array has been executed and the output is verified.

EX .NO: 10 MATRIX OPERATIONS

DATE:

AIM:

To perform matric addition, subtraction and Multiplication.

ALGORITHM:

Step 1: Start the Program.

Step 2: Declare three 2dimensional array variable[a,b,c].

Step 3: Get the dimensions for the matrix .

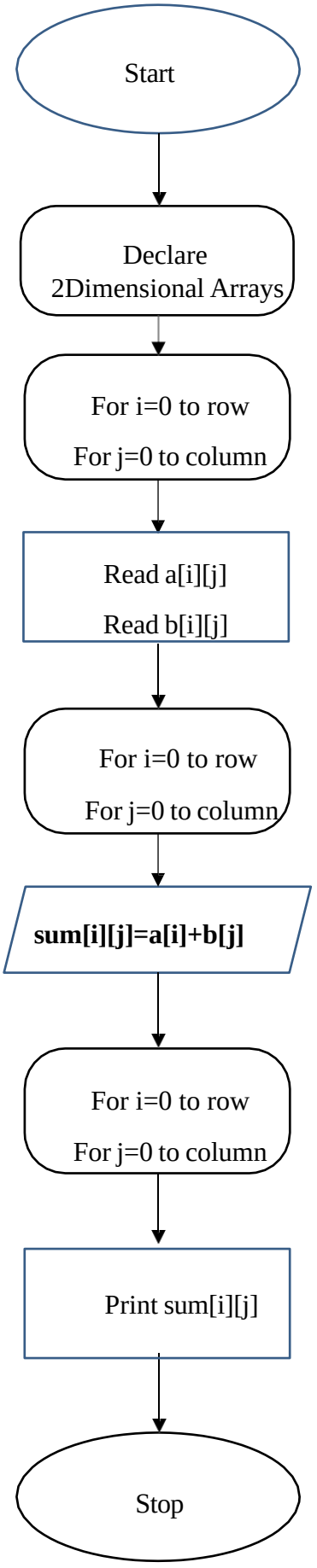
Step 4: Get the values of the two matrices [a and b]using for-loop.

Step 5: The sum, difference and multiplication of the two matrices to be stored in an other sum, diff and mul arrays.

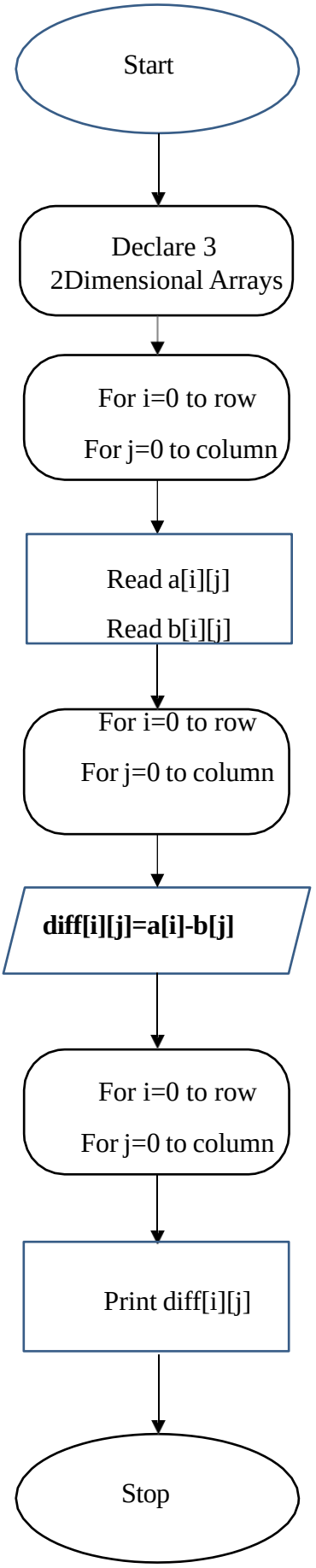
Step 6: Print the matrix other sum, diff and mul arrays by using for-loop.

Step 7: Stop the program.

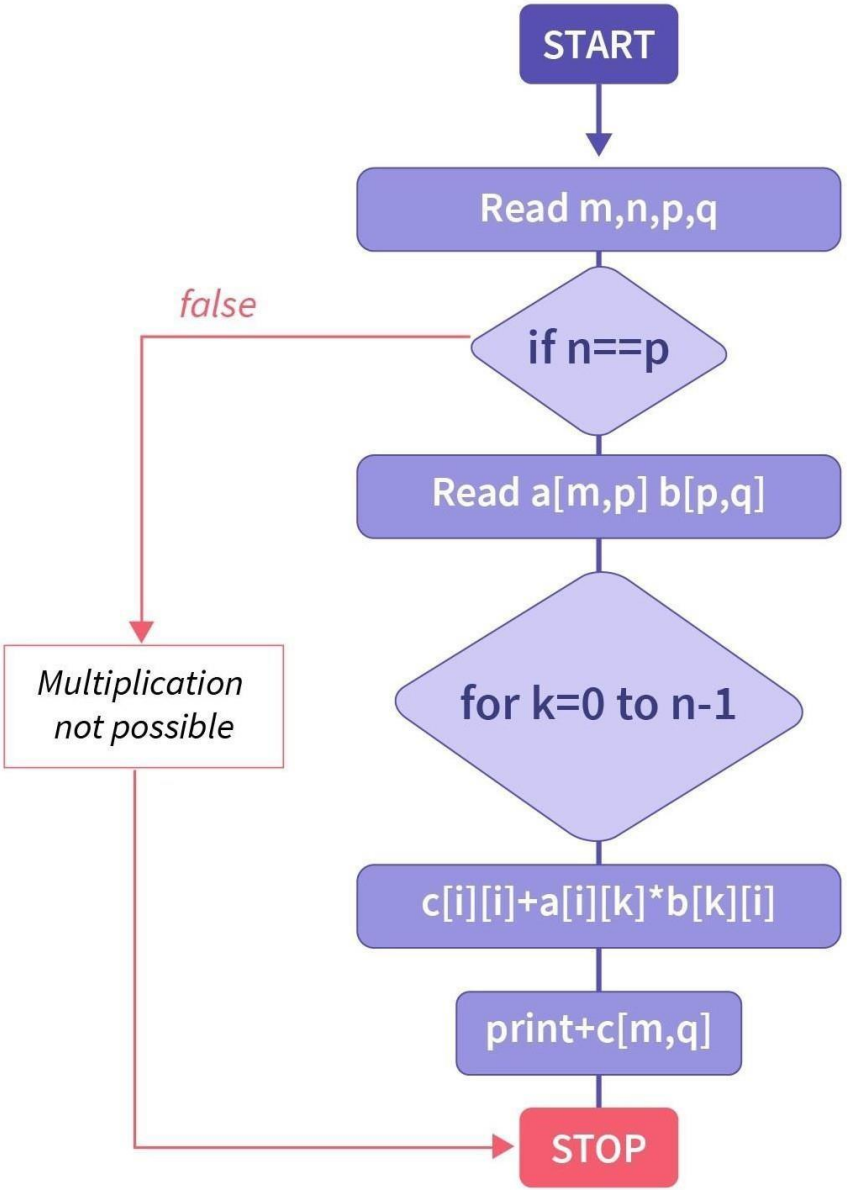
FLOWCHART: **ADDITION**



SUBTRACTION



MULTIPLICATION



PROGRAM:

/* To perform matric addition*/

```
#include<stdio.h>
int main()
{
	int i, j, m, n, a[10][10], b[10][10], sum[10][10], diff[10][10], mul[10][10], k;
	printf("\nEnter the number of rows and columns of the first matrix \n\n");
	scanf("%d %d", &m,&n);
	printf("\nEnter the %d elements of the first matrix \n\n",m*n);
	for(i = 0; i < m; i++)
	for(j = 0; j< n; j++)
	scanf("%d", &a[i][j]);
	printf("\nEnter the %d elements of the second matrix \n\n", m*n);
	for(i = 0; i < m; i++)
	for(j = 0; j< n; j++) scanf("%d", &b[i][j]);
	printf("\n\nThe first matrix is: \n\n"); for(i = 0; i< m; i++)
	{
		for(j = 0; j < n; j++)
		{
			printf("%d\t", a[i][j]);
		}
	printf("\n");
	}
	printf("\n\nThe second matrix is: \n\n"); for(i= 0; i< m; i++)
	{
		for(j = 0; j < n; j++)
	{
	printf("%d\t", b[i][j]);
	}
	printf("\n");
	}
		for(i = 0; i< m; i++)
		for(j =0; j < n; j++)
		sum[i][j] = a[i][j] + b[i][j];
		printf("\n\nThe sum of the two entered matrices is: \n\n"); for(i = 0; i < m; i++)
	{
```

```
        for(j = 0; j< n; j++)
        {
                printf("%d\t", sum[i][j]);
        }
printf("\n");
}
for(i = 0; i < m; i++)
{
for(j = 0; j < n; j++) diff[i][j] = a[i][j] - b[i][j];
}
printf("\n\nThe difference(subtraction) of the two entered matrices is: \n\n");
for(i = 0; i < m; i++)
{
        for(j = 0; j < n; j++)
        {
                printf("%d\t", diff[i][j]);
        }
printf("\n");
}
printf("Multiply of the matrix:\n"); for(i=0;i<m;i++)
{
        for(j=0;j<n;j++)
        {
                 mul[i][j]=0; for(k=0;k<n;k++)
                {
                mul[i][j]+=a[i][k]*b[k][j];
                }
        } }
        for(i=0;i<m;i++)
{
        for(j=0;j<n;j++)
{
        printf("%d\t",mul[i][j]);
}
printf("\n");
}       return 0;   }
```

OUTPUT:

```
Enter the number of rows and columns of the first matrix

2
2

Enter the 4 elements of the first matrix

2
3
1
1

Enter the 4 elements of the second matrix

1
2
3
4


The first matrix is:

2       3
1       1


The second matrix is:

1       2
3       4


The sum of the two entered matrices is:

3       5
4       5


The difference(subtraction) of the two entered matrices is:

1       1
-2      -3
Multiply of the matrix:
11      16
4       6
```

Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to implement matrix addition, subtraction and multiplication has been executed and the output is verified.

EX .NO: 11.A LINEAR SEARCH

DATE:

AIM:

To write a C-program to search an element in an array using linear searching.

ALGORITHM:

Step 1: Start the program

Step 2: Declare the variables to get an array and the search element

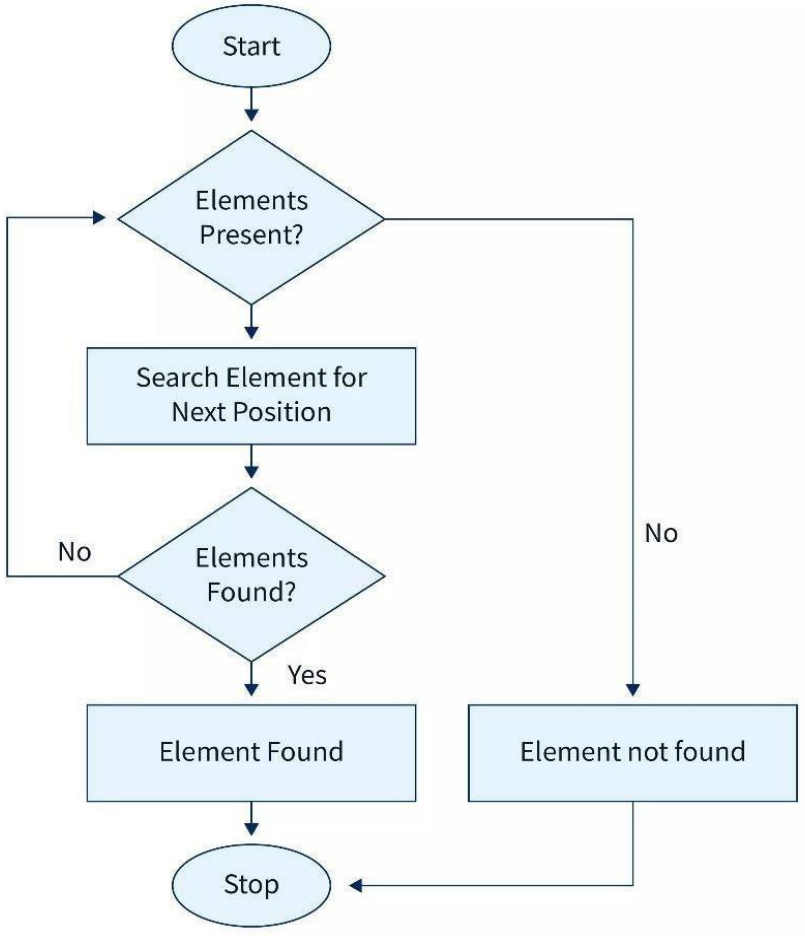
Step 3: Check whether the search element is present in the array

Step 4: If present , print the index at which it is present

Step 5: If not, print that the element is not found

Step 6: Stop the program.

FLOWCHART:

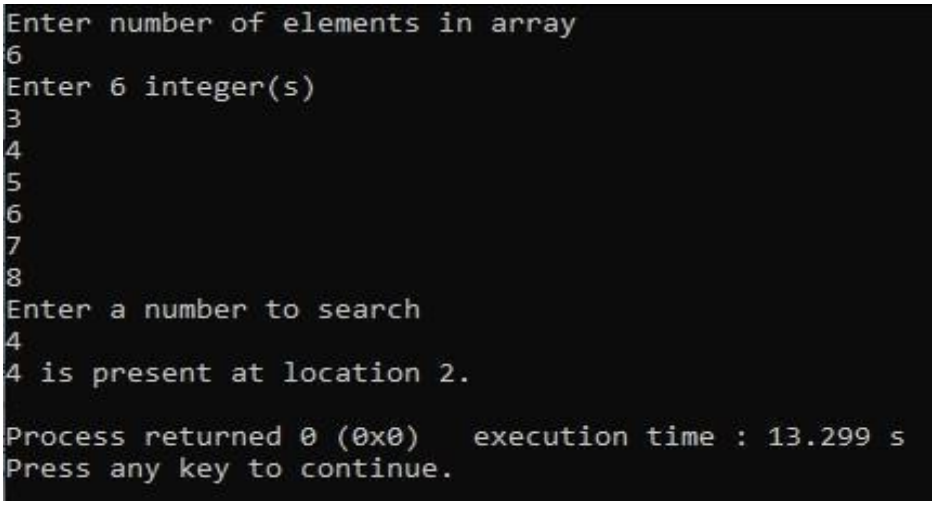


PROGRAM:

/* To write a C-program to search an element in an array using linear searching*/

```
#include <stdio.h>
int main()
{
  int a[10], key, i, n;
  printf("Enter number of elements in array\n");
  scanf("%d", &n);
  printf("Enter %d integer(s)\n", n);
  for (i = 0; i < n; i++)
   {
     scanf("%d", &a[i]);
   }
  printf("Enter a number to search\n");
  scanf("%d", &key);
  for (i = 0; i < n; i++)

       {
    if (a[i] == key)
    {
      printf("%d is present at location %d.\n", key, i+1);
      break;
    }
  }
  if (i == n)
    printf("%d isn't present in the array.\n", key);
  return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the C program to search the given number using linear search is executed and the output is verified.

EX .NO: 11.B BINARY SEARCH

DATE:

AIM:

To write a C-program to search an element in an array using binary searching.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare the variables to get an array and the search element.

Step 3: The array has to be sorted.

Step 4: Take the middle element and check whether the search element is greater than or lesser than or equal to the middle element.

Step 5: If greater, change the starting element as middle element.

Step 6: If lesser , change the last element as middle element.

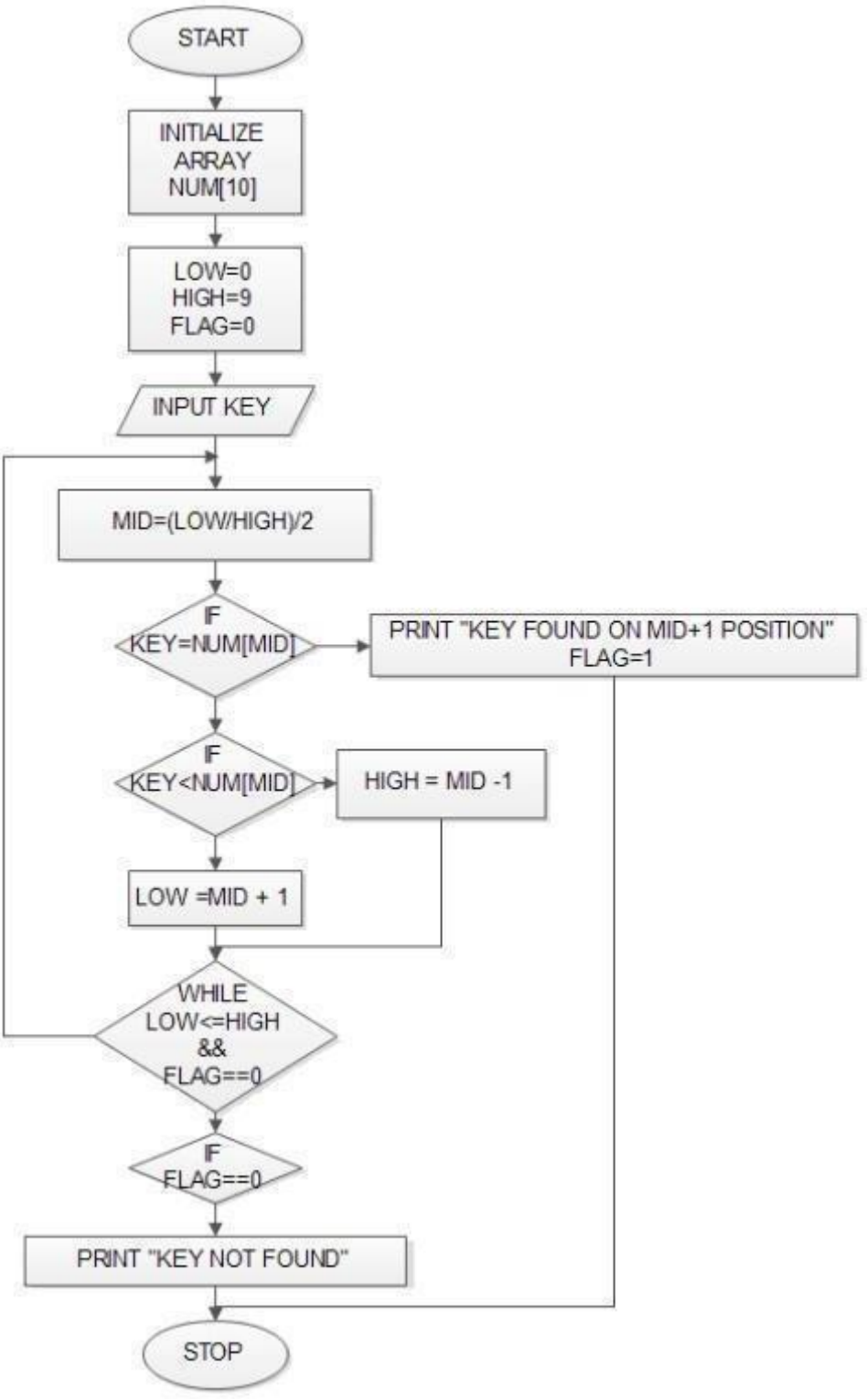
Step 7: And then find the middle element for it and repeat the process.

Step 8: If equal, print that the element is found at this index.

Step 9: If not, print that the element is not found.

Step 10: Stop the program.

FLOWCHART:


START
INITIALIZE ARRAY NUM[10]
LOW=0 HIGH=9 FLAG=0
INPUT KEY
MID=(LOW/HIGH)/2
IF KEY=NUM[MID]
PRINT "KEY FOUND ON MID+1 POSITION" FLAG=1
IF KEY<NUM[MID]
HIGH = MID -1
LOW =MID + 1
WHILE LOW<=HIGH && FLAG==0
IF FLAG==0
PRINT "KEY NOT FOUND"
STOP

PROGRAM:

/* To write a C-program to search an element in an array using binary searching*/

```
#include <stdio.h>
int main()
{
  int i, first, last, middle, n, key, a[100];
  printf("Enter number of elements\n");
  scanf("%d", &n);
  printf("Enter %d integers\n", n);
  for (i = 0; i < n; i++)
   {
    scanf("%d", &a[i]);
   }
  printf("Enter value to find\n");
  scanf("%d", &key);
  first = 0;
  last = n - 1;
  middle = (first+last)/2;
  while (first <= last)
{
    if (a[middle] < key)
      {
        first = middle + 1;
      }
```

```
    else if (a[middle] == key)
     {
       printf("%d found at location %d.\n", key, middle+1);
       break;
     }
    else
      last = middle - 1;
    middle = (first + last)/2;
  }
  if (first > last)
    printf("Not found! %d isn't present in the list.\n", key);
  return 0;
}
```

OUTPUT:

```
Enter number of elements
5
Enter 5 integers
34
2
5
67
89
Enter value to find
67
67 found at location 4.

Process returned 0 (0x0)   execution time : 13.439 s
Press any key to continue.
```

Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program is to search the given array number by using binary search is executed and the output is verified.

EX .NO: 12.A BUBBLE SORT

DATE:

AIM:

To sort an unsorted numbers of array using bubble sort.

ALGORITHM:

Step 1: Start the program

Step 2: Declaring the variables

Step 3: Starting with the first element (index=0) , compare the current element with the next element of the array

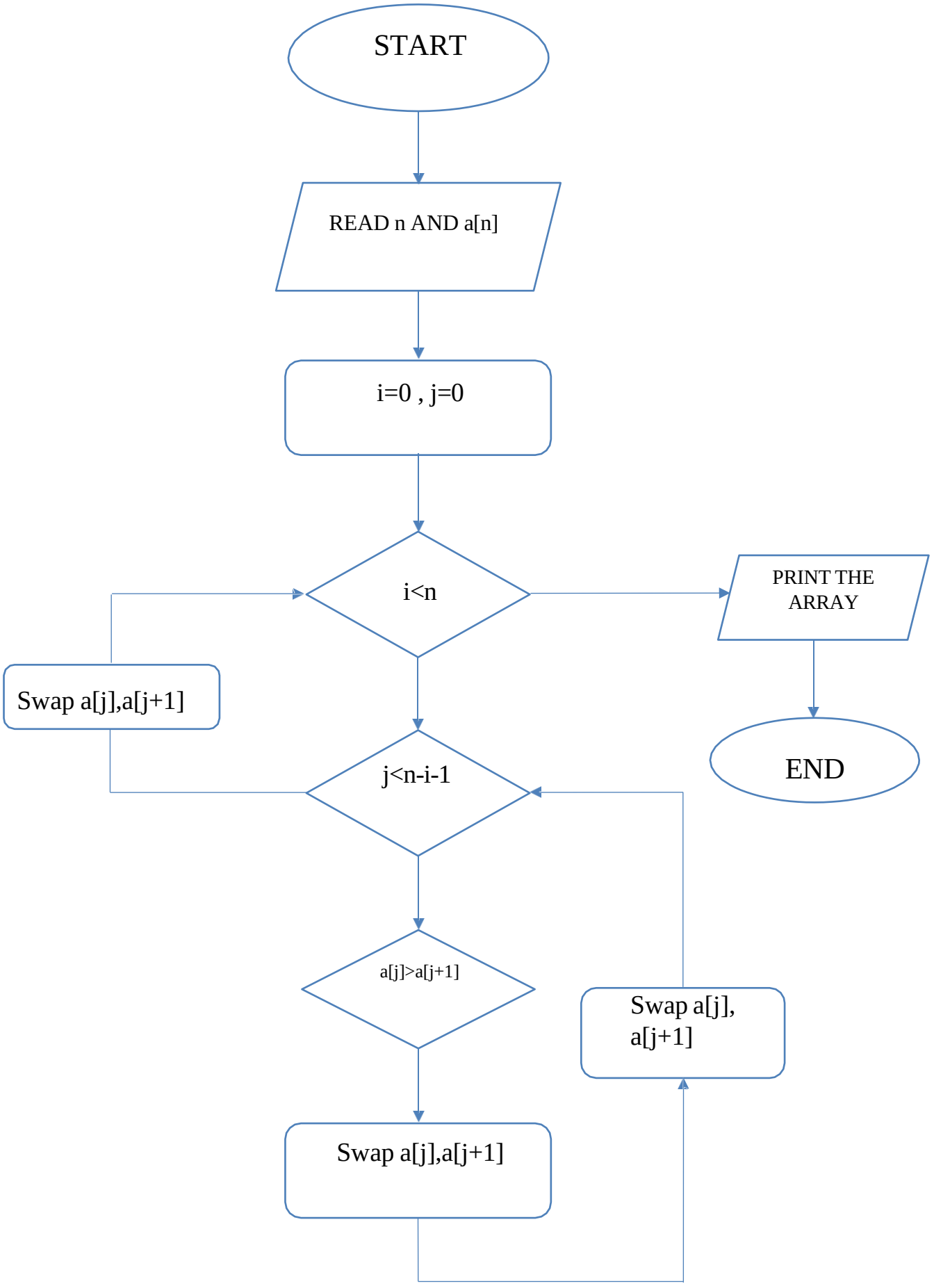
Step 4: If the current element is greater than the next element of the array, swap them

Step 5: If the current element is less than the next element, move to the next element

Step 6: Repeat Step 1

Step 7: Stop the program.

FLOWCHART:


START
READ n AND a[n]
i=0 , j=0
i<n
PRINT THE ARRAY
END
Swap a[j],a[j+1]
j<n-i-1
a[j]>a[j+1]
Swap a[j], a[j+1]
Swap a[j],a[j+1]

PROGRAM:

/* To sort an unsorted numbers of array using bubble sort*/

```
#include <stdio.h>
int main()
{
  int array[100], n, i, j, swap;
  printf("Enter number of elements\n");
  scanf("%d", &n);
  printf("Enter %d integers\n", n);
  for (i = 0; i < n; i++)
  scanf("%d", &array[i]);
  for (i = 0 ; i < n - 1; i++)
  {
    for (j = 0 ; j < n - i - 1; j++)
    {
      if (array[j] > array[j+1])
      {
        swap       = array[j];
        array[j]  = array[j+1];
        array[j+1] = swap;
      }
    }
  }
  printf("Sorted list in ascending order:\n");
```

```
  for (i = 0; i < n; i++)
    printf("%d\n", array[i]);
  return 0;
}
```

OUTPUT:

```
Enter number of elements
7
Enter 7 integers
56
45
32
21
0
32
4
Sorted list in ascending order:
0       4       21      32      32      45      56
Process returned 0 (0x0)   execution time : 13.486 s
Press any key to continue.
```

Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to sort the given set of number by using bubble sort is executed and the output was verified.

EX .NO: 12.B SELECTION SORT

DATE:

AIM:

To sort an unsorted array of numbers using selection sort.

ALGORITHM:

Step 1: Start the program

Step 2: Declaring the variables

Step 3: Get the input array values

Step 4: Set min to the first location

Step 5: Search the minimum element in the array

Step 6: Swap the first location with the minimum value in the array

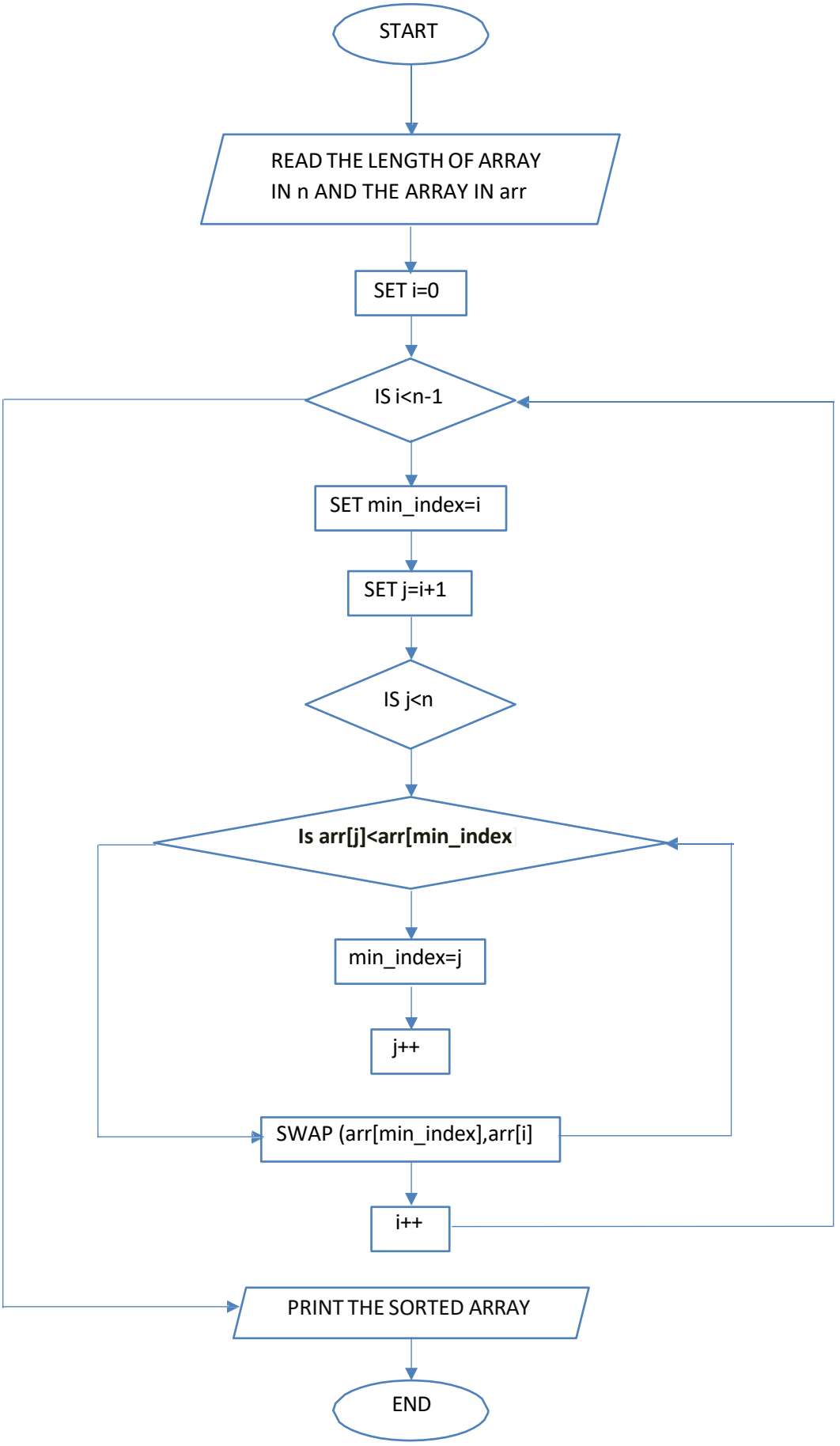
Step 7: Assign the second element as min.

Step 8: Repeat the process until we get a sorted array.

Step 9: Print the result

Step 10: Stop the program.

FLOWCHART:


START
READ THE LENGTH OF ARRAY
IN n AND THE ARRAY IN arr
SET i=0
IS i<n-1
SET min_index=i
SET j=i+1
IS j<n
Is arr[j]<arr[min_index
min_index=j
j++
SWAP (arr[min_index],arr[i]
i++
PRINT THE SORTED ARRAY
END

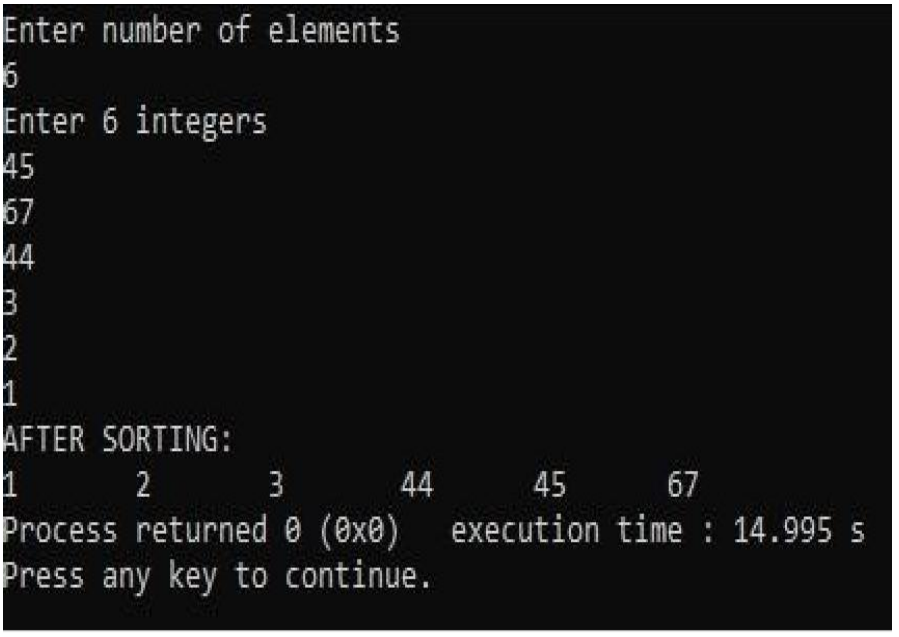
PROGRAM:

To sort an unsorted array of numbers using selection sort

```
#include <stdio.h>

int main()

{

  int array[100], n,p,i,j, t;

  printf("Enter number of elements\n");

  scanf("%d", &n);

  printf("Enter %d integers\n", n);

  for (i = 0; i < n; i++)

  {

    scanf("%d", &array[i]);

}

  for (i = 0; i < (n - 1); i++)

  {

     p=i;

    for (j = i + 1; j < n; j++)

    {

     if (array[p] > array[j])

    {

       p=j;

    }

    if(p!=i)

    {

      t = array[i];
```

```
      array[i] = array[p];
      array[p] = t;
    }
    }
  }
  printf("AFTER SORTING:\n");
  for (i = 0; i < n; i++)
  printf("%d\t", array[i]);
    return 0;
  }
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program is sorting the given set of array number by using selection sort has been executed and the output is verified.

EX .NO: 12.C INSERTION SORT

DATE:

AIM:

To sort an unsorted array of numbers using insertion sort.

ALGORITHM:

Step 1: Start the program

Step 2: Declare the variables

Step 3: Read the variables

Step 4: If the element is the first one, it is already sorted

Step 5: Move to next element

Step 6: Compare the current element with all elements in the sorted array

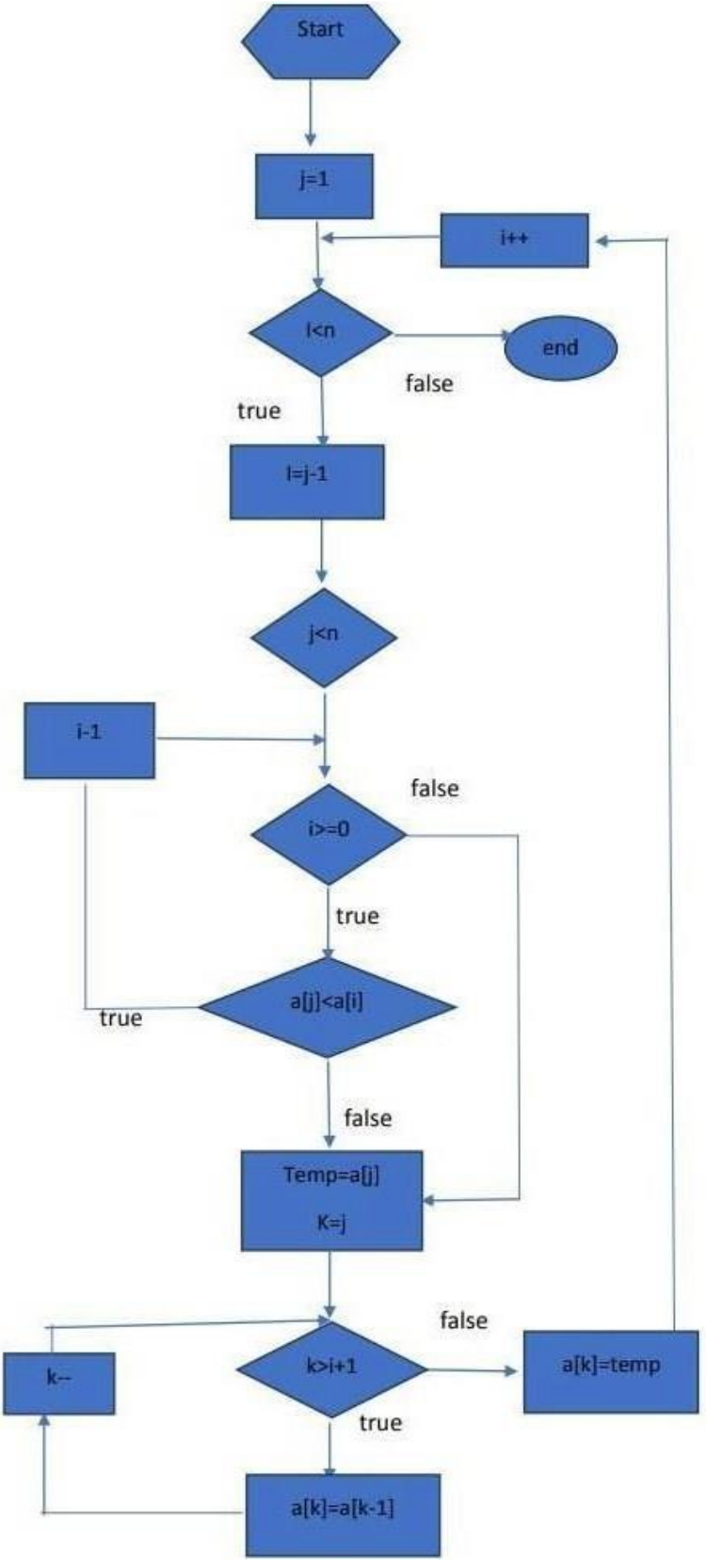
Step 7: If the element in the sorted array is smaller than the current element, iterate to the next element. Otherwise, shift all the greater element in the array by one position towards the right

Step 8: Insert the value at the correct position

Step 9: Repeat until the complete list is sorted

Step 10: Stop the program

FLOWCHART:


Start
j=1
i++
I<n
end
false
true
I=j-1
j<n
i-1
false
i>=0
true
a[j]<a[i]
true
false
Temp=a[j]
K=j
false
k--
k>i+1
a[k]=temp
true
a[k]=a[k-1]

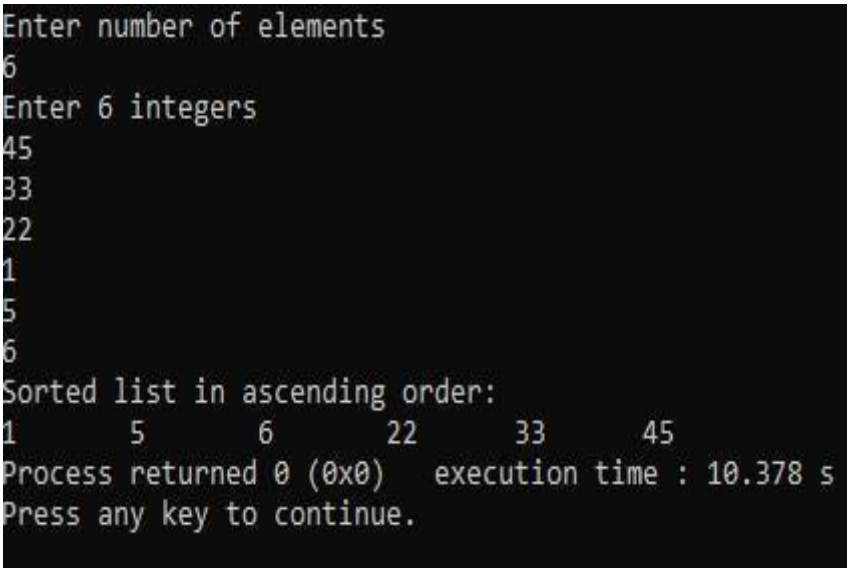
PROGRAM:

/* To sort an unsorted array of numbers using insertion sort*/

```
#include <stdio.h>
int main()
{
  int n, array[1000], i, j, t, flag = 0;
  printf("Enter number of elements\n");
  scanf("%d", &n);
  printf("Enter %d integers\n", n);
  for (i = 0; i < n; i++)
  scanf("%d", &array[i]);
  for (i = 1 ; i <= n - 1; i++)
 {
    t = array[i];
    for (j = i - 1 ; j >= 0; j--)
   {
      if (array[j] > t)
      {
        array[j+1] = array[j];
        flag = 1;
      } else
      break;
    }
```

```
    if (flag)
      array[j+1] = t;
  }
  printf("Sorted list in ascending order:\n");
  for (i = 0; i <= n - 1; i++)
  {
    printf("%d\t", array[i]);
  }
  return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to sort the given set of number by using insertion sort has been executed and the output is verified.

EX.NO: 13 FUNCTION AND RECURSION

DATE:

AIM:

To find the factorial of a given number using Function and Recursion.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare one variables.

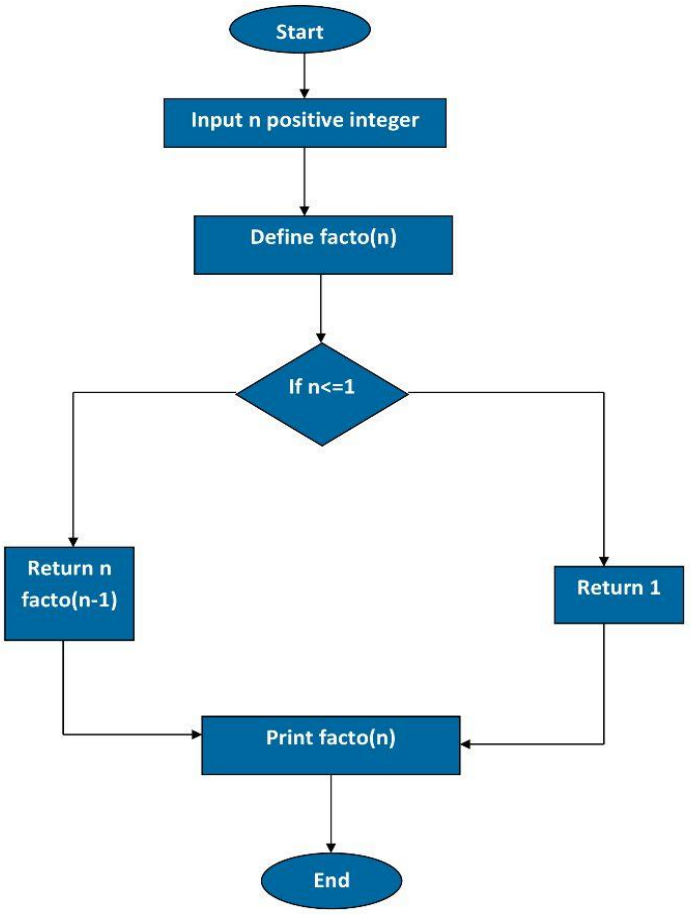
Step 3: Using function , given number is even number or odd number .

Step 4: Using recursive function , 2 to the power of given number .

Step 5: Print the output to verify.

Step 6: Stop the program.

FLOWCHART:

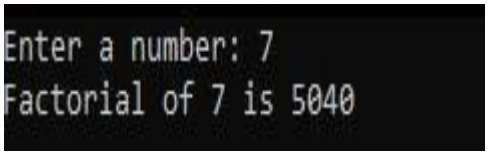


PROGRAM:

/*To find the factorial of a given number using Function and Recursion*/

```
#include<stdio.h>

long factorial(int n)

{

  if (n == 0)

  return     1;

  else

    return(n * factorial(n-1));

}
void main()

{

  int number;

  long fact;

  printf("Enter a number: ");

  scanf("%d", &number);

  fact = factorial(number);

  printf("Factorial of %d is %ld\n", number, fact);

  return 0;

}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to implement factorial of n numbers using recursive function has been executed and the output is verified.

EX.NO. :14 CALL BY VALUE AND CALL BY REFERENCE

DATE:

AIM:

To swap two numbers using call by value and call by reference.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare two variables.

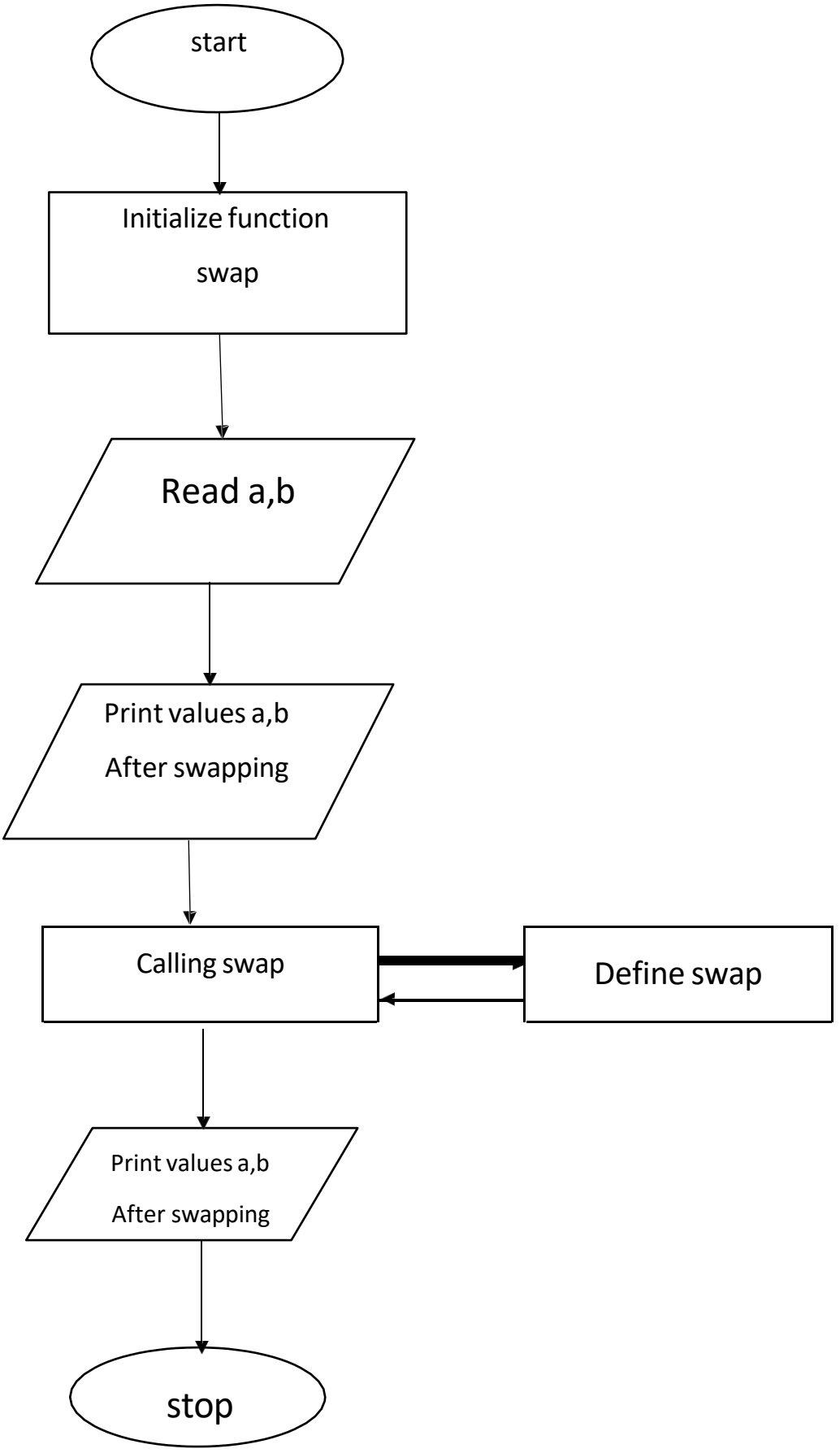
Step 3: Using call by value method swap between the two numbers.

Step 4: Using call by reference method, swap the value between the two numbers.

Step 5: Print the output to verify.

Step 6: Stop the program.

FLOWCHART:


start
Initialize function
swap
Read a,b
Print values a,b
After swapping
Calling swap
Define swap
Print values a,b
After swapping
stop

PROGRAM:

//Call by value

```
#include<stdio.h>
void swap(int a, int b);
int main()
{
   int m,n;
   printf("Enter the values of m and n\n");
   scanf("%d %d",&m,&n);
   printf(" values before swap  m = %d and n = %d", m, n);
   swap(m, n);
return 0;
}
void swap(int a, int b)
{
   int  tmp;
   tmp  =  a;
   a = b;
   b = tmp;
   printf(" \nvalues after swap  m = %d and n = %d", a,b);
}
//Call by reference
#include <stdio.h>

void swap(int a, int
*b);

int main()
{
   int a = 10;
   int b = 20;
   printf("Before swapping the values in main a = %d, b = %d\n",a,b);
   swap(&a,&b);


}
```

```
void swap (int *a, int *b)
{
   int temp;
   temp = *a;
   *a=*b;
   *b=temp;
   printf("\n After swapping values in function a = %d, b = %d\n",*a,*b);
}
```

OUTPUT:

Call by value:

```
Enter the values of m and n
50
43
 values before swap  m = 50 and n = 43
values after swap  m = 43 and n = 50
Process returned 0 (0x0)   execution time : 7.994 s
Press any key to continue.
```

Call by reference:

```
Before swapping the values in main a = 10, b = 20

 After swapping values in function a = 20, b = 10

Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program for call by value and call by reference has been executed and the output is verified.

Exp. No: 15 SUM OF ELEMENTS IN AN ARRAY USING POINTERS

DATE:

AIM:

To write a C program to find the sum of the integer array using pointers.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare the variables.

Step 3: Initialize the array elements with values from user input.

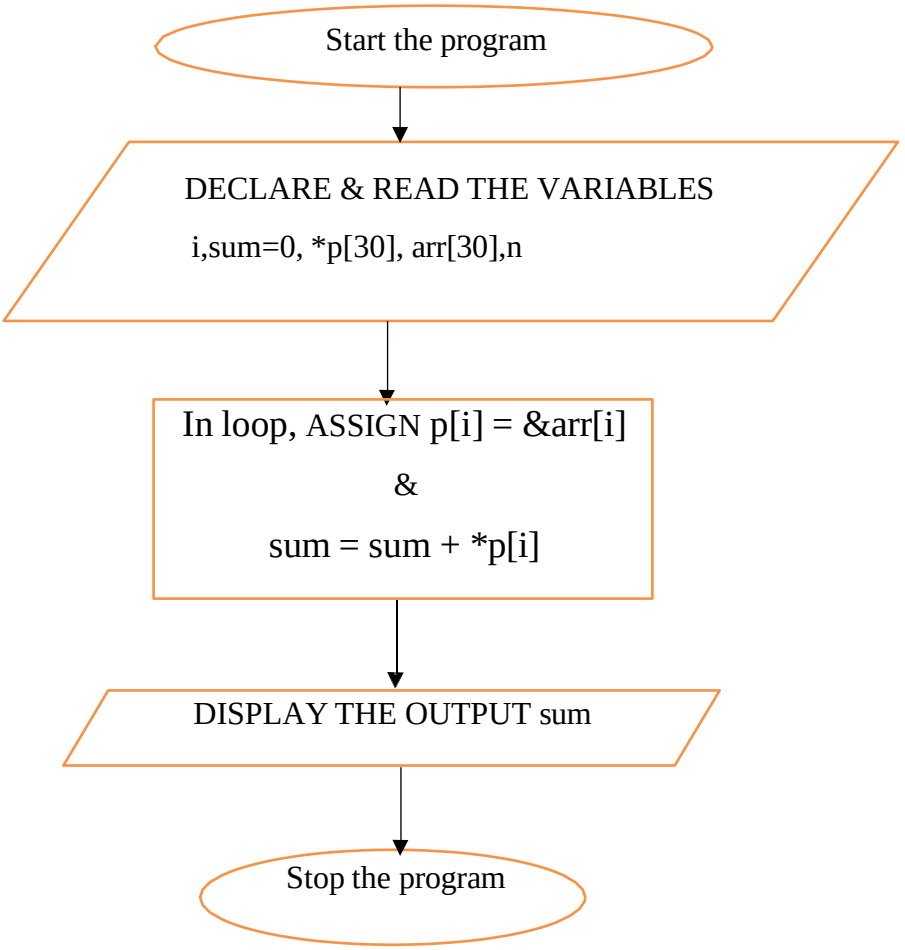
Step 4: Initialize sum = 0.

Step 5: Loop for i = 0 to n-1.

Step 6: Display the output (sum value).

Step 7: Stop the program.

FLOWCHART:

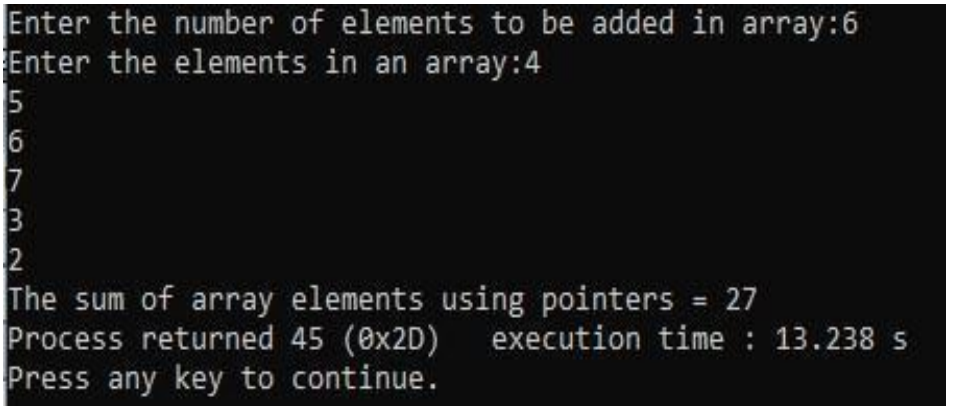

Start the program
DECLARE & READ THE VARIABLES
i,sum=0, *p[30], arr[30],n
In loop, ASSIGN p[i] = &arr[i]
&
sum = sum + *p[i]
DISPLAY THE OUTPUT sum
Stop the program

PROGRAM:

/* To write a C program to find the sum of the integer array using pointers*/

```
#include  <stdio.h>
#include <conio.h>
void main()
{
   int i,sum=0, *p[30], arr[30],n;
   printf("Enter the number of elements to be added in array:");
   scanf("%d",&n);
   printf("Enter the elements in an array:");
   for(i=0;i<n;i++)
   {
      p[i] = &arr[i];
      scanf("%d,&arr[i]");
   }
   for(i=0;i<n;i++)
   {
      sum = sum + *p[i] ;
   }
   printf("The sum of array elements using pointers = %d",sum);
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to find the sum of the integer array using pointers has been executed and output is verified.

Exp. No:16 MAXIMUM ELEMENT IN AN ARRAY USING POINTERS

DATE:

AIM:

To write a C program to find the maximum element in an integer array using pointers.

ALGORITHM:

Step 1: Start the program.

Step 2: Declare the variables**.**

Step 3: Initialize the array elements with values from user input.

Step 4: Initialize max = 0.

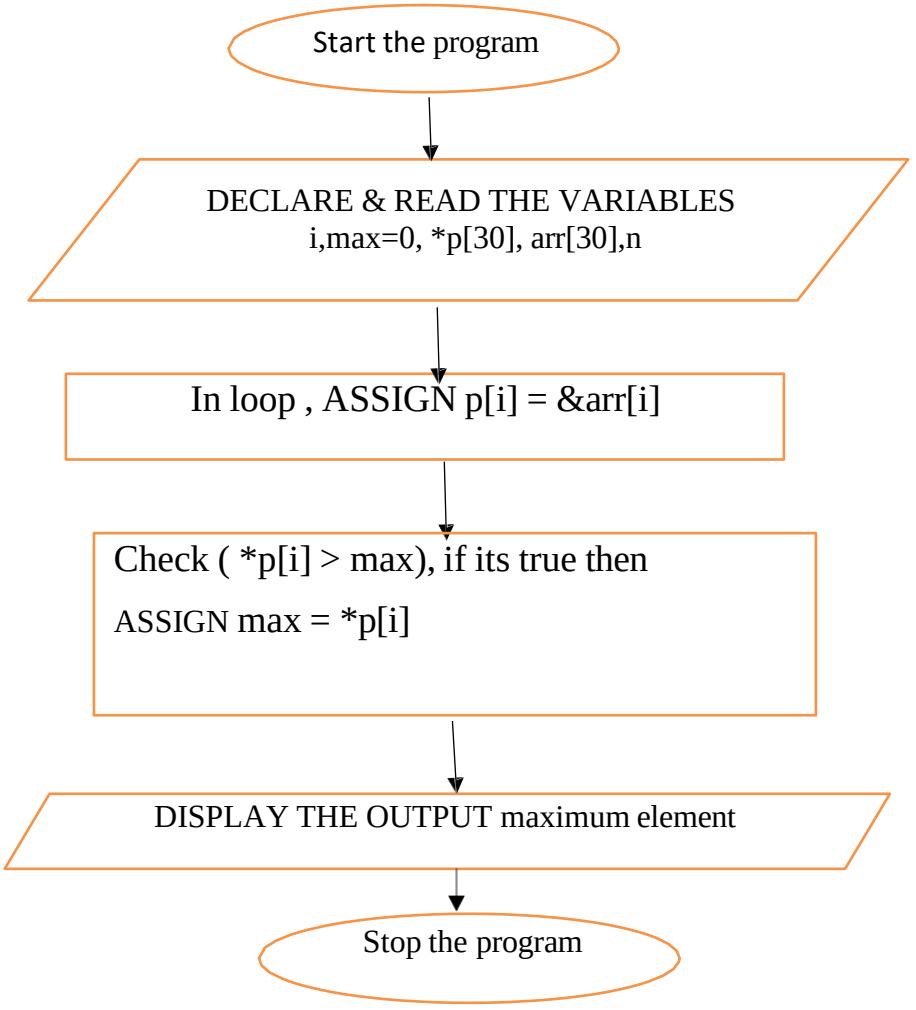
Step 5: Loop for i = 0 to n-1.

Step 6: check *p[i] > max, If it is true then, max=*p[i].

Step 6: Display the output (maximum element).

Step 7: Stop the program.

FLOWCHART:

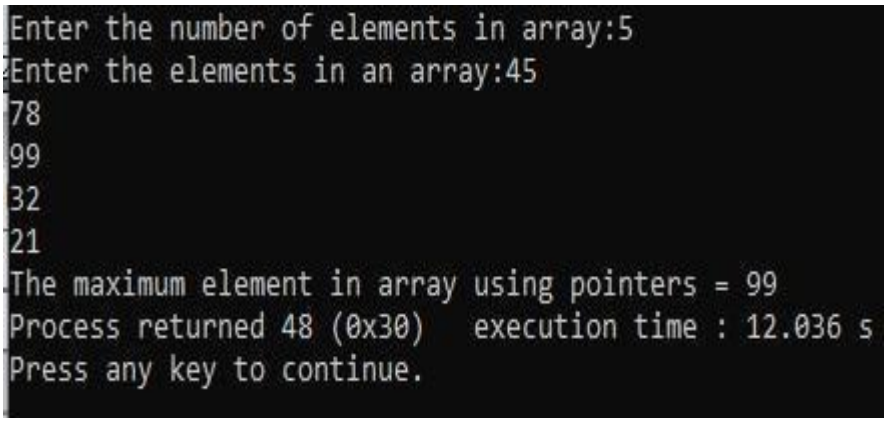

Start the program
DECLARE & READ THE VARIABLES
i,max=0, *p[30], arr[30],n
In loop , ASSIGN p[i] = &arr[i]
Check (*p[i] > max), if its true then
ASSIGN max = *p[i]
DISPLAY THE OUTPUT maximum element
Stop the program

PROGRAM:

/* To write a C program to find the maximum element in an integer array using pointers*/

```
#include  <stdio.h>

#include <conio.h>

void main()

{

   int i,max=0, *p[30], arr[30],n;

   printf("Enter the number of elements  in array:");

   scanf("%d",&n);

   printf("Enter the elements in an array:");

   for(i=0;i<n;i++)

   {

      p[i] = &arr[i];

      scanf("%d,&arr[i]");

   }

   for(i=0;i<n;i++)

   {

      if( *p[i] > max)

      {

         max = *p[i] ;

      }

   }

   printf("The maximum element in array using pointers = %d",max);

}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the C program to find the maximum element in an integer array using pointers has been executed and output is verified.

EX.NO. :17 GENERATE SALARY SLIP OF EMPLOYEES USING STRUCTURE AND POINTERS

DATE:

AIM:

To generate salaryslip of employees using structure and pointers.

ALGORITHM:

Step 1: start the program

Step 2: declare the structure and some variables to collect the employee details

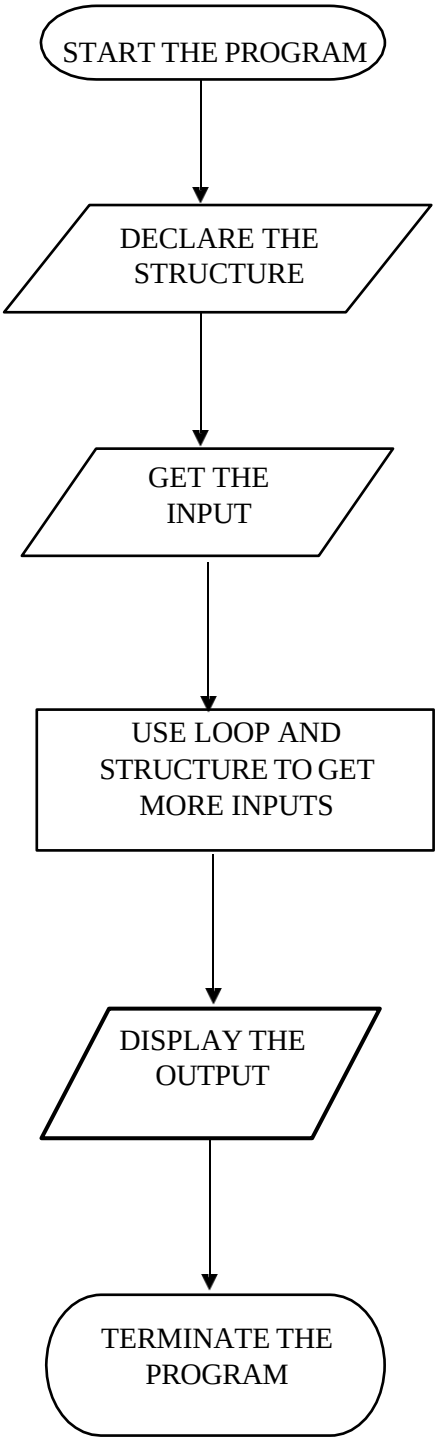
Step 3: get values for the declared variables

Step 4: use structure and loop to get more number of details

Step 5: print the output

Step 6: terminate the program

FLOW CHART:


START THE PROGRAM
DECLARE THE STRUCTURE
GET THE INPUT
USE LOOP AND STRUCTURE TO GET MORE INPUTS
DISPLAY THE OUTPUT
TERMINATE THE PROGRAM

PROGRAM:

/* To generate salary slip of employees using structure and pointers*/

```
#include <stdio.h>
struct SalarySlip
{
int no;
int empid;
char Name[60];
char Job_type[100];
int Salary;
};
int add_row(struct SalarySlip *row,int size);
int display_row(struct SalarySlip *row,int size);
int main()
{
int n;
printf("Enter the no. of rows :");
scanf("%d",&n);
struct SalarySlip S1[n],*ptr;
ptr=&S1[0];
add_row(ptr,n);
display_row(ptr,n);

return 0;
}
int add_row(struct SalarySlip *row,int size)
{
for (int i=0;i<size;i++)
{
printf("\nEnter Serial no :");
```

```
scanf("%d",&row[i].no);

printf("\nEnter employee id :");
scanf("%d",&row[i].empid);

printf("\nEnter name :");
scanf("%s",row[i].Name)
;

printf("\nEnter job type :");
scanf("%s",row[i].Job_type);

printf("\nEnter Salary :");
scanf("%d",&row[i].Salary);
}

return 0;
}

int display_row(struct SalarySlip *row,int size)
{
for (int i=0;i<size;i++)
{
printf("\nSI.no=%d \nEMP.no=%d \nName=%s \nJob Type=%s
\nSalary=Rs.%d\n",row[i].no,row[i].empid,row[i].Name,row[i].job_type,row[i].Salary);
}
return 0;
}
```

OUTPUT:

```
Enter the no. of rows :2

Enter Serial no :123

Enter employee id :4566

Enter name :Sakthi

Enter job type :HR

Enter Salary :34000

Enter Serial no :145

Enter employee id :678

Enter name :Praveen

Enter job type :Manager

Enter Salary :35000

SI.no=123
EMP.no=4566
Name=Sakthi
Job Type=HR
Salary=Rs.34000

SI.no=145
EMP.no=678
Name=Praveen
Job Type=Manager
Salary=Rs.35000
```

Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, c program to generate salary slip of employees using structure and pointers is executed.

EX.NO:18 TO DISPLAY THE CONTENT OF A FILE IN A MONITOR SCREEN

DATE:

AIM:

To display the content of a file in a monitor screen.

ALGORITHM:

Step 1: Start the program

Step 2: declare the file pointer

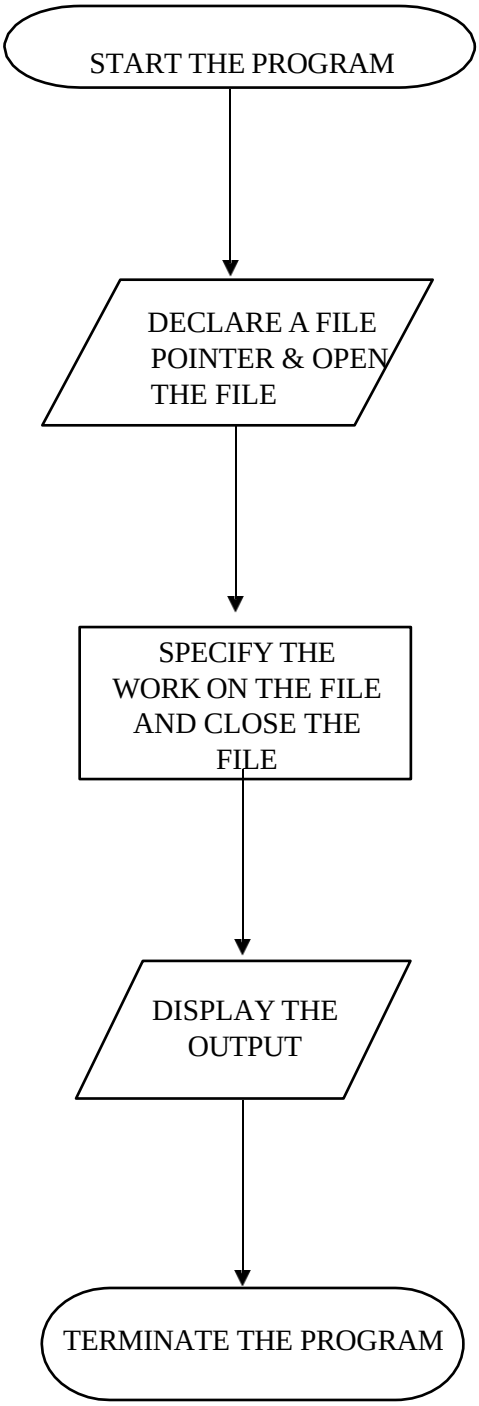
Step 3: open the file in the program

Step 4: specify the mode to work on it

Step 5: close the opened file

Step 6: Terminate the program

FLOW CHART:

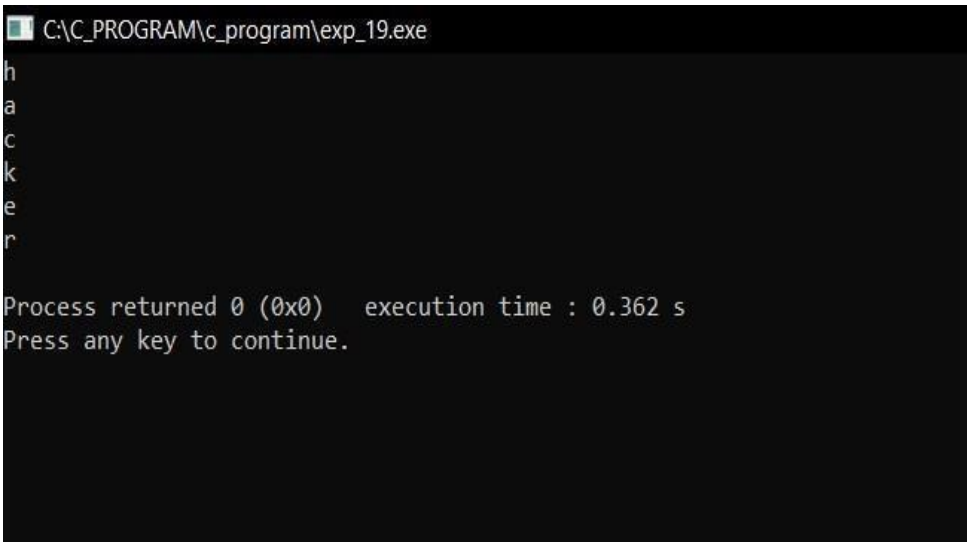

START THE PROGRAM
DECLARE A FILE POINTER & OPEN THE FILE
SPECIFY THE WORK ON THE FILE AND CLOSE THE FILE
DISPLAY THE OUTPUT
TERMINATE THE PROGRAM

PROGRAM:

/* To display the content of a file in a monitor scree*/

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

int main()
{
   FILE * fptr;
   char ch;
   fptr = fopen("sample.txt", "r");
   if(fptr == NULL)
   {
       printf("File does not exist.");

   }
   else
   {
      while(fscanf(fptr,"%c", &ch)!= EOF)
      {
         printf("%c", ch);
      }
   }
    return 0;

}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the program to display the content of a file in a monitor screen has been executed and the output is verified.

EX. No : 19 GETTING THE INPUT FROM THE KEYBOARD AND RETRIEVE THE CONTENTS OF THE FILE USING FILE OPERATION COMMANDS

DATE:

AIM:

To Get the input from the keyboard and retrieve the contents of the file using file operation commands

ALGORITHM:

STEP 1: Start the program

STEP 2: Open the file for reading using fopen.

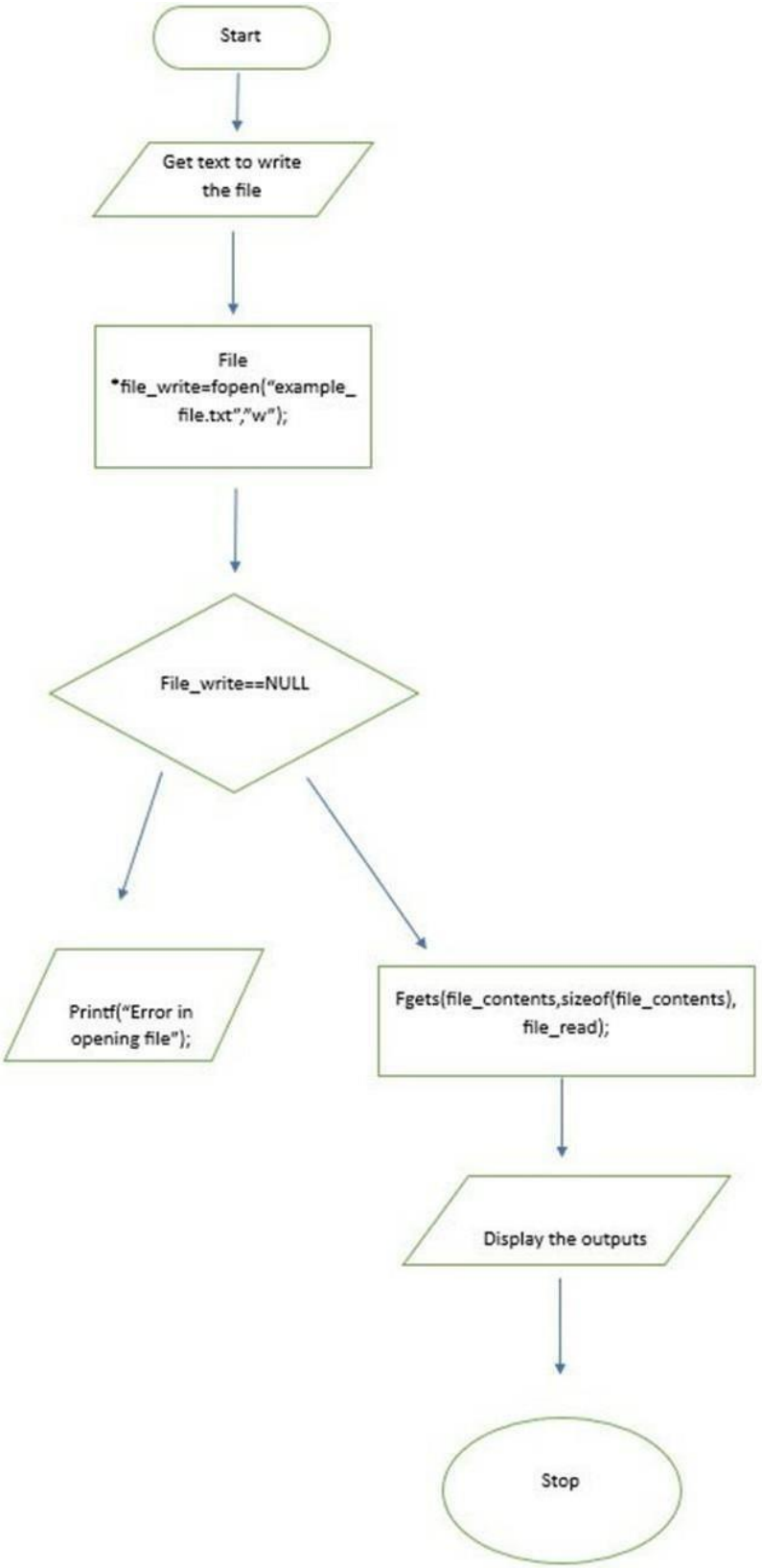
STEP 3: Check if the file was opened successfully.
If not, print an error message and exit the program.

STEP 4: Use a loop to read and print each character until the end of the file (EOF) is reached.

STEP 5: Close the file using fclose.

STEP 6:End the program

FLOW CHART:


Start
Get text to write the file
File *file_write=fopen("example_ file.txt","w");
File_write==NULL
Printf("Error in opening file");
Fgets(file_contents,sizeof(file_contents), file_read);
Display the outputs
Stop

PROGRAM:

*** Getting the input from the keyboard and retrieve the contents of the file using file operation commands ***

```
#include <stdio.h>

int main() {
// Open a file for reading
FILE *file = fopen("example.txt", "r");

// Check if the file was opened successfully
if (file == NULL) {
printf("Error opening the file.\n");
return 1; // Exit the program with an error code
}

// Print the contents of the file to the console
printf("Contents of the file:\n");

// Read and print each character until the end of the file is reached
char ch;
while ((ch = fgetc(file)) != EOF) {
putchar(ch);
}

// Close the file
fclose(file);

// Take input from the keyboard
printf("\nEnter something from the keyboard: ");
char userInput[100];
fgets(userInput, sizeof(userInput), stdin);

// Print the input from the keyboard
printf("You entered: %s\n", userInput);

return 0;
}
```

OUTPUT:

```
Enter text to write to the file: Learn C programs and all other
Contents of the file:
Learn C programs and all other
```

Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus the c program to implement To Get the input from the keyboard and retrieve the contents of the file using file operation commands has been executed and the output is verified.

EX. No : 20 TO CREATE TWO FILES WITH A SET OF VALUES MERGE THE TWO FILE CONTENTS TO FORM A SINGLE FILE

DATE:

AIM:

To create two files with a set of values. Merge the two file contents to form a single file.

ALGORITHM:

STEP 1:Start the program

STEP 2: Open the first file for reading using fopen.

STEP 3: Open a new file (destination file) for writing using fopen. This file will store the merged content.

STEP 4: Check if all file openings were successful.
If not, print an error message and exit the program.

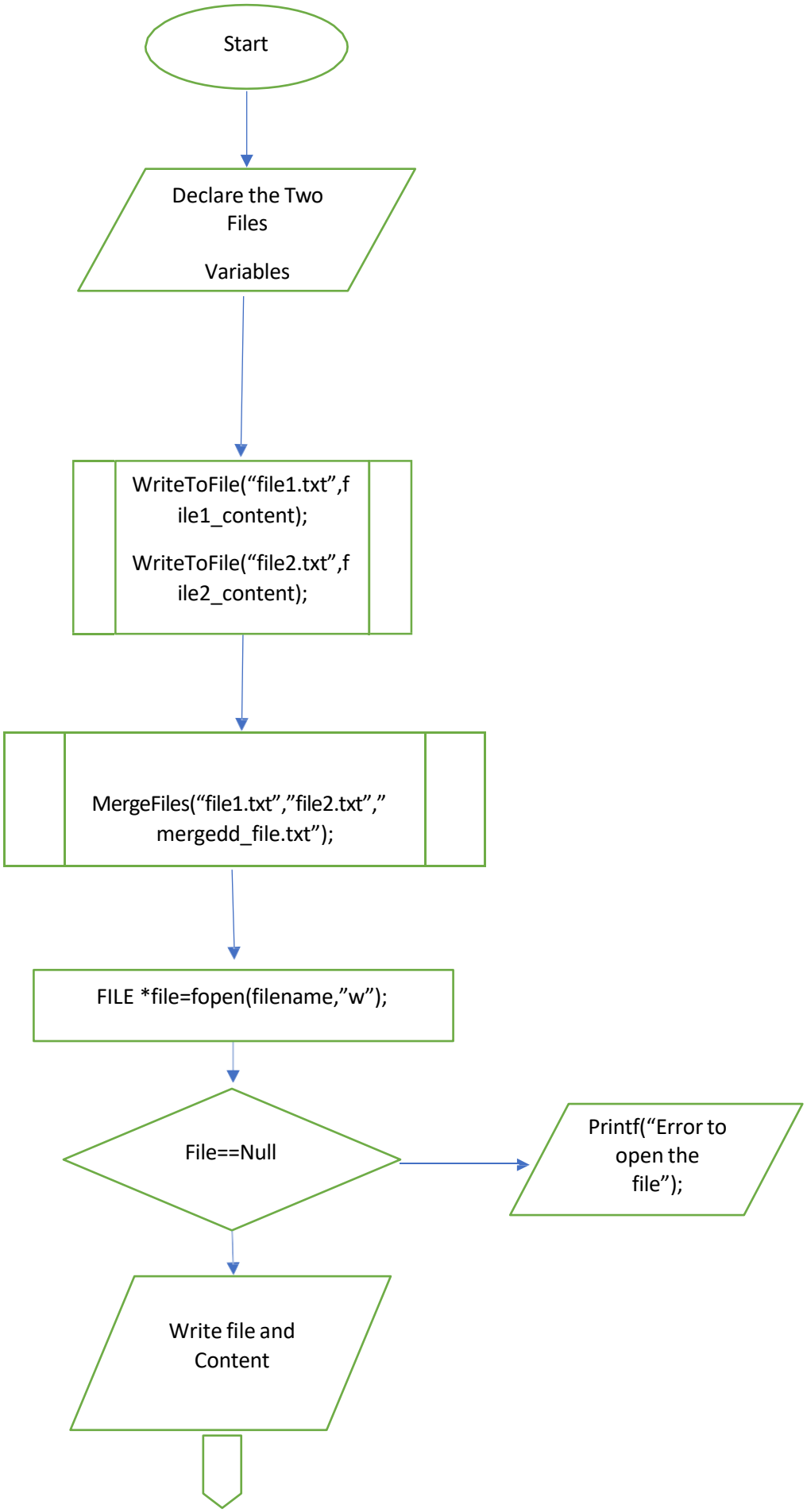
STEP 5: Read each character from the first file and write it to the destination file.

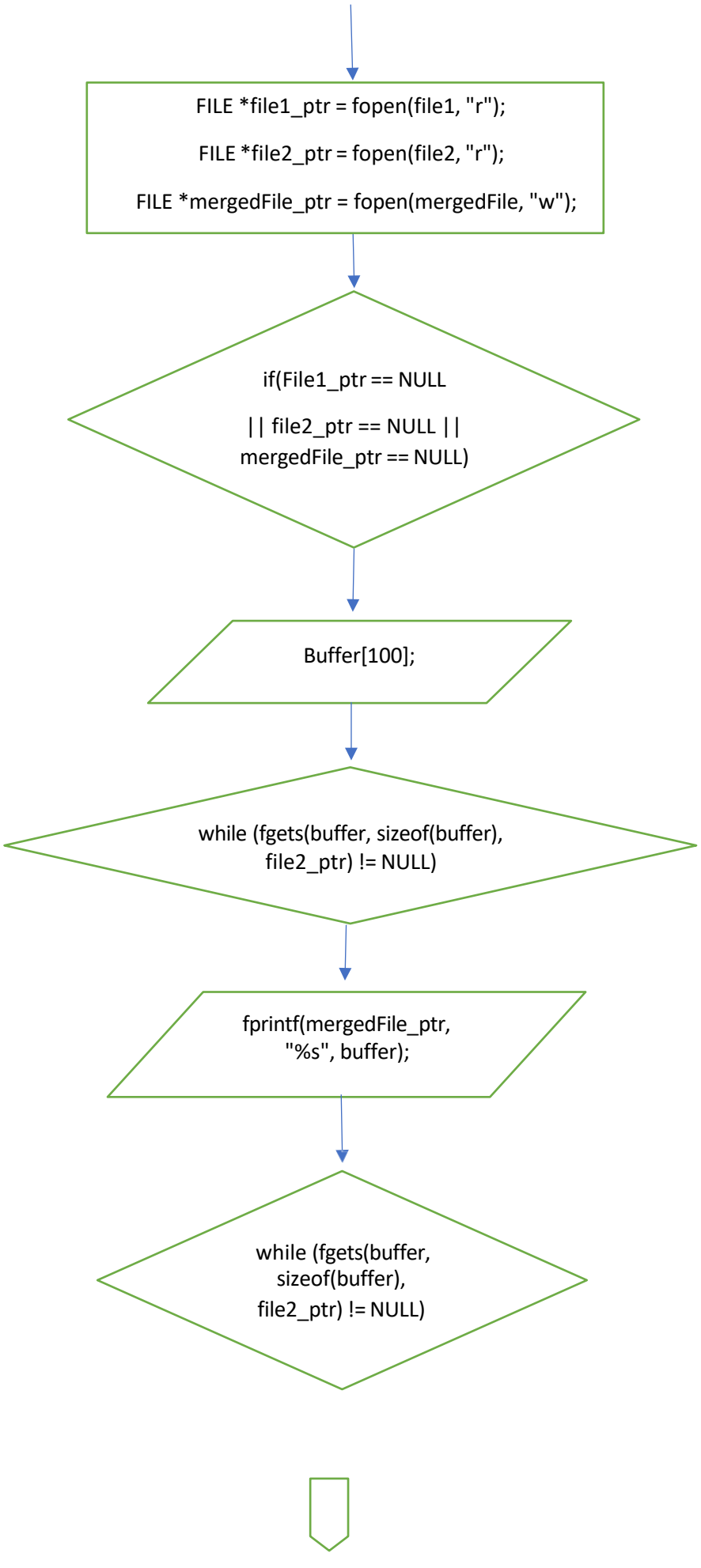
STEP 6: Read each character from the second file and write it to the destination file.

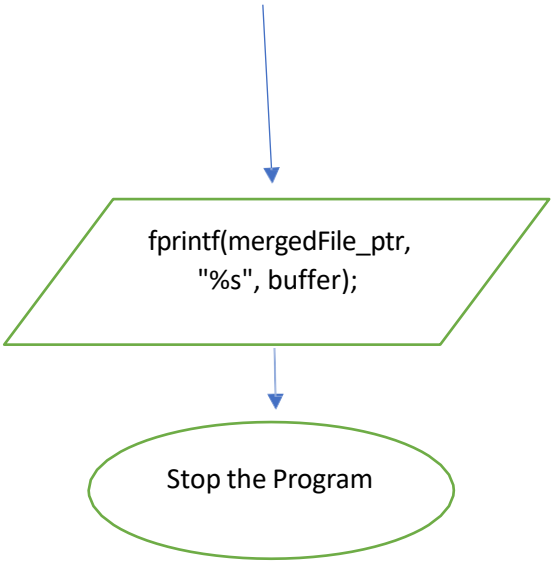
STEP 7: Close all the opened files.

STEP 8: End the program.

FLOWCHART:


Start
Declare the Two Files
Variables
WriteToFile("file1.txt",file1_content);
WriteToFile("file2.txt",file2_content);
MergeFiles("file1.txt","file2.txt","mergedd_file.txt");
FILE *file=fopen(filename,"w");
File==Null
Printf("Error to open the file");
Write file and Content


FILE *file1_ptr = fopen(file1, "r");
FILE *file2_ptr = fopen(file2, "r");
FILE *mergedFile_ptr = fopen(mergedFile, "w");
if(File1_ptr == NULL
|| file2_ptr == NULL ||
mergedFile_ptr == NULL)
Buffer[100];
while (fgets(buffer, sizeof(buffer),
file2_ptr) != NULL)
fprintf(mergedFile_ptr,
"%s", buffer);
while (fgets(buffer,
sizeof(buffer),
file2_ptr) != NULL)


fprintf(mergedFile_ptr,
"%s", buffer);
Stop the Program

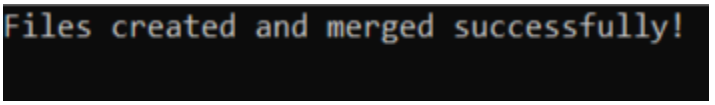
PROGRAM:

*** To create two files with a set of values. Merge the two file contents to form a single file ***

```
#include <stdio.h>

int main() {
    // Open the first file for reading
    FILE *file1 = fopen("file1.txt", "r");

    // Open the second file for reading
    FILE *file2 = fopen("file2.txt", "r");

    // Open the destination file for writing
    FILE *mergedFile = fopen("merged_file.txt", "w");

    // Check if file openings were successful
    if (file1 == NULL || file2 == NULL || mergedFile == NULL) {
        printf("Error opening files.\n");
        return 1; // Exit the program with an error code
    }

    // Read and write contents of the first file
    char ch;
    while ((ch = fgetc(file1)) != EOF) {
        fputc(ch, mergedFile);
    }

    // Read and write contents of the second file
    while ((ch = fgetc(file2)) != EOF) {
        fputc(ch, mergedFile);
    }

    // Close all opened files
    fclose(file1);
    fclose(file2);
```

```
    fclose(mergedFile);

    return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus the c program To create two files with a set of values. Merge the two file contents to form a single file has been executed and the output is verified.

Ex. No:21 COMMAND LINE ARGUMENTS

DATE:

AIM:

To pass the parameter using Command line Arguments

ALGORITHM:

Step 1: Start the program.

Step 2: Declare parameters for main() function.

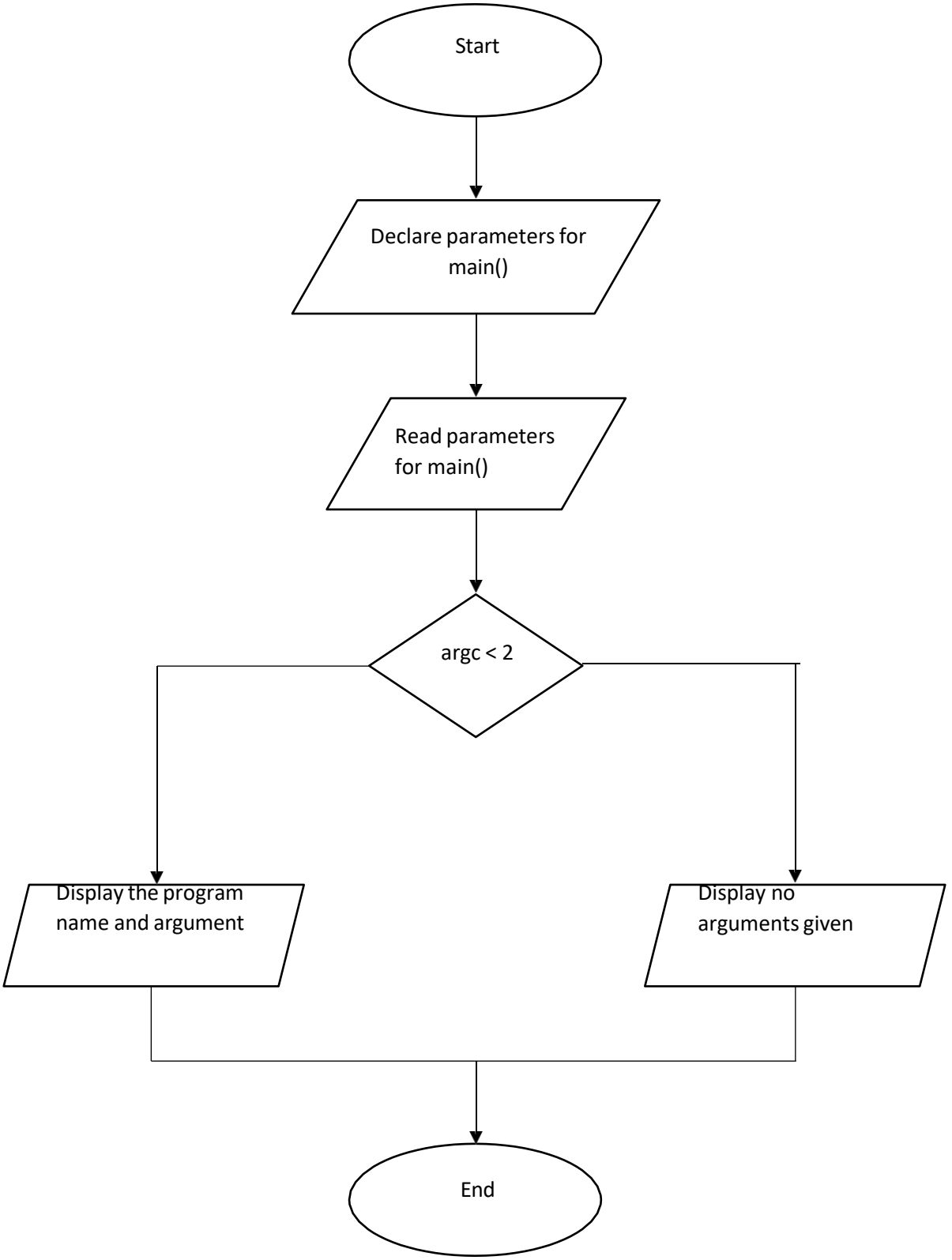
Step 3: Read parameters for main() function.

Step 4: If the correct number of arguments are given, display program name and argument.

Step 5: else display no arguments given.

Step 6: End the program.

FLOW CHART:

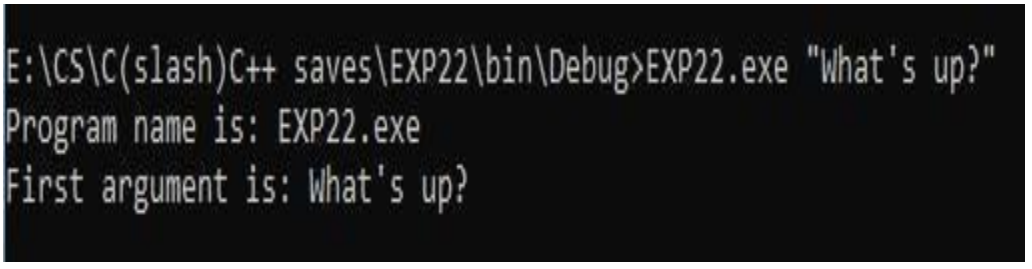

Start
Declare parameters for main()
Read parameters for main()
argc < 2
Display the program name and argument
Display no arguments given
End

PROGRAM:

/* Command line arguments*/

```
#include <stdio.h>

void main(int argc, char *argv[])

{

   printf("Program name is: %s\n", argv[0]);

   if(argc < 2)

   {

      printf("No argument passed through command line.\n");

   }

   else

   {

      printf("First argument is: %s\n", argv[1]);

   }

}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program to implement finding the total and average marks is executed and the output is verified.

EX. No : 22 **FACTORIAL USING FUNCTIONS**

DATE :

AIM:

To find the factorial of a given number using Functions in C.

ALGORITHM:

STEP 1:Start the program

STEP 2: Define a function factorial that takes an integer parameter n and returns an integer.
If n is 0 or 1, return 1 (base case).
Otherwise, return n * factorial(n - 1) (recursive case).

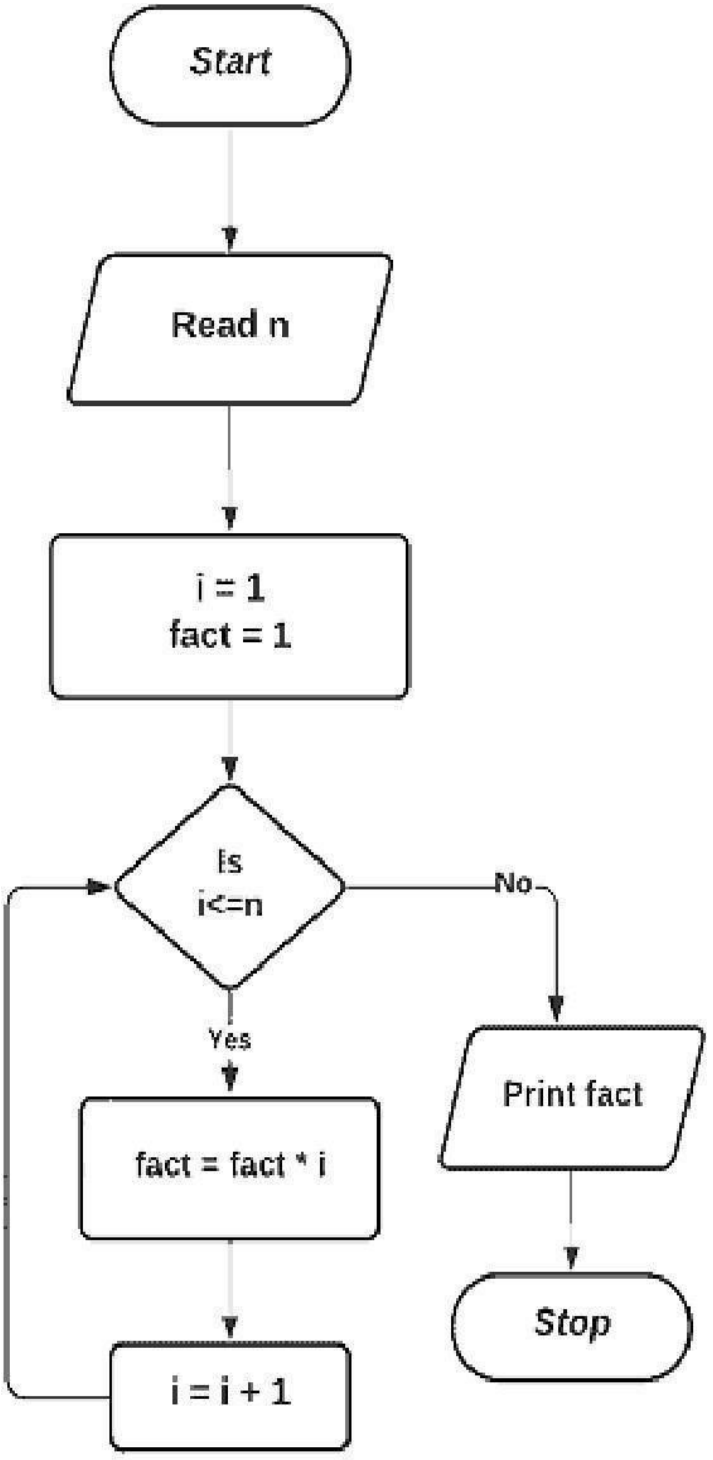
STEP 3: In the main function:
Declare a variable num to store the user input.
Input: Prompt the user to enter a number and store it in num

STEP 4: Check if the entered number is negative.
If true, print an error message
If false, proceed to the next step.

STEP 5: Call the factorial function with the entered number as an argument.
Print the result.

STEP 6: End the program.

FLOW CHART

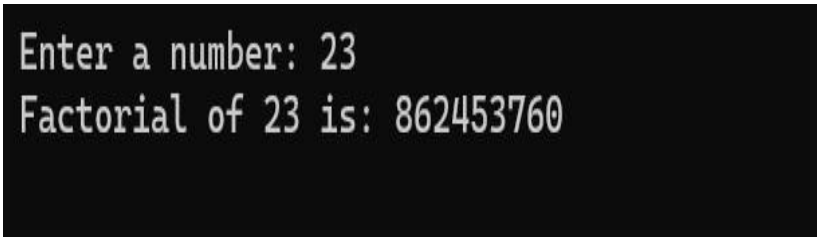

Start
Read n
i = 1
fact = 1
Is
i<=n
No
Yes
Print fact
fact = fact * i
Stop
i = i + 1

PROGRAM:

*** To find the factorial of a given number using Functions in C.***

```
#include <stdio.h>

// Function to calculate the factorial of a number
int factorial(int n) {
if (n == 0 || n == 1) {
return 1; // Factorial of 0 and 1 is 1
} else {
return n * factorial(n - 1);
}
}

int main() {
int num;

// Input: Get the number from the user
printf("Enter a number: ");
scanf("%d", &num);

// Check for negative numbers
if (num < 0) {
printf("Factorial is not defined for negative numbers.\n");
} else {
// Calculate and print the factorial using the factorial function
printf("Factorial of %d is: %d\n", num, factorial(num));
}

return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus the c program To find the factorial of a given number using Functions in C has been executed and the output is verified.

EX. No : 23 PALINDROME OR NOT

DATE :

AIM:

To check whether the given string is Palindrome or not.

ALGORITHM:

STEP 1:Start the program

STEP 2: Ask the user to enter a string

.

STEP 3 :Set length to 0 (to count the length of the string).

Set flag to 1 (assume the string is a palindrome initially).

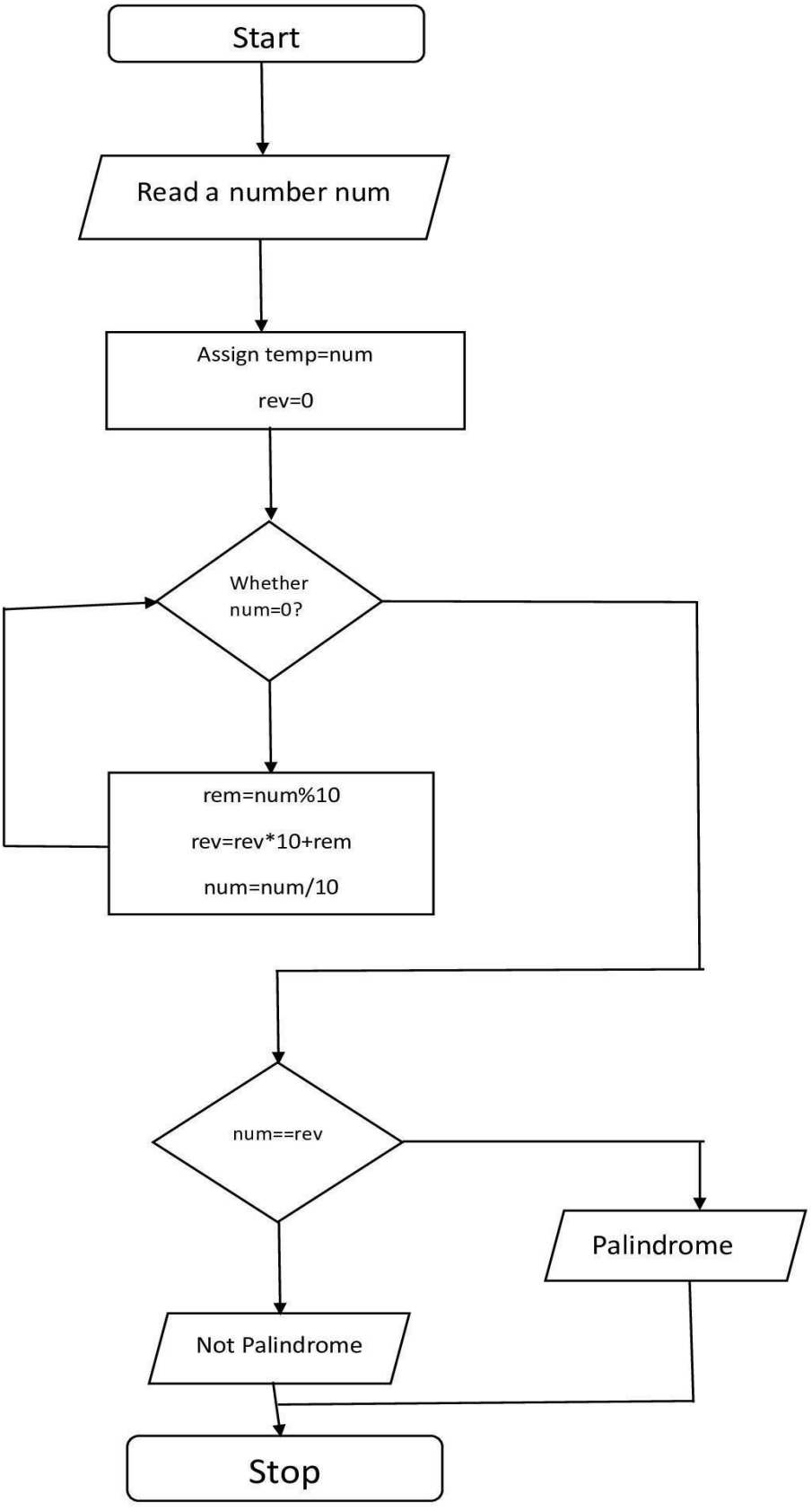
STEP 4: Use a loop to go through each character of the string and count the length

STEP 5: Use another loop to compare characters from the start and end of the string.

If the characters are not equal, set flag to 0 and break out of the loop.

STEP 6: End the program.

FLOW CHART:

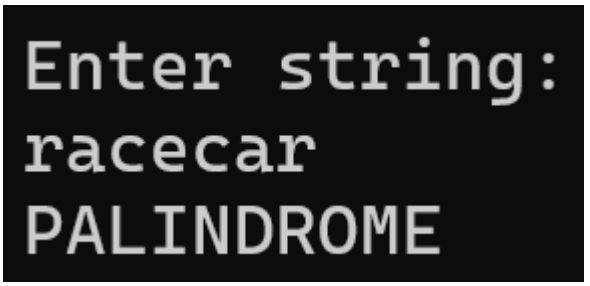

Start
Read a number num
Assign temp=num
rev=0
Whether
num=0?
rem=num%10
rev=rev*10+rem
num=num/10
num==rev
Palindrome
Not Palindrome
Stop

PROGRAM:

To check whether the given string is Palindrome or not.

```
#include<stdio.h>

int main()
{
   char string[40];
   int length=0, flag=1,i;

   printf("Enter string:\n");
   gets(string);

   for(i=0;string[i]!='\0';i++)
   {
      length++;
   }

   for(i=0;i< length/2;i++)
   {
      if( string[i] != string[length-1-i] )
      {
         flag=0;
         break;
      }
   }

   if(flag==1)
   {
      printf("PALINDROME");
   }
   else
   {
      printf("NOT PALINDROME");
   }
   return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus the c program To check whether the given string is Palindrome or not has been executed and the output is verified.

EX. No : 24 PRIME OR NOT

DATE :

AIM:

To check whether the given string is Prime or not.

ALGORITHM:

STEP 1: Start the program

STEP 2: Set i to 2 for the loop iteration.
Set flag to 0 (assume the number is prime initially).

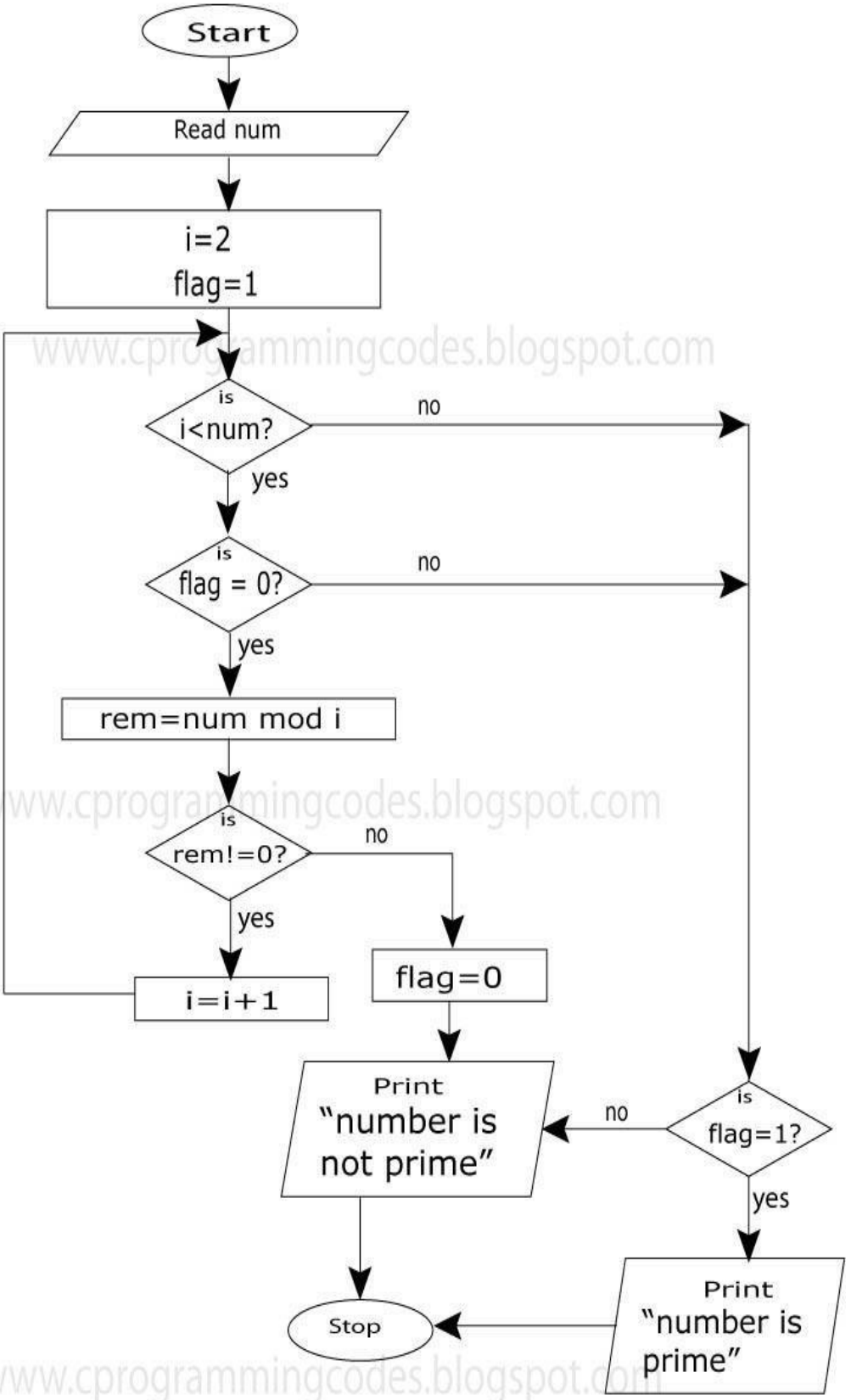
STEP 3 : If the entered number is 0 or 1, set flag to 1 (non-prime).

STEP 4: Use a loop to check divisibility from 2 to half of the entered number.
If the entered number is divisible by any number in the loop, set flag to 1 and .
break out of the loop

STEP 5: Print whether the entered number is prime or not based on the value of flag.

STEP 6: End the program.

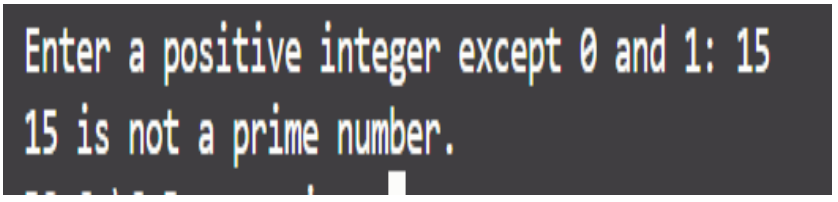
FLOW CHART:


Start
Read num
i=2
flag=1
is
i<num?
no
yes
is
flag = 0?
no
yes
rem=num mod i
is
rem!=0?
no
yes
i=i+1
flag=0
Print
"number is
not prime"
is
flag=1?
no
yes
Print
"number is
prime"
Stop
www.cprogrammingcodes.blogspot.com

PROGRAM:

```
\* To check whether the given string is Prime or not. *\

#include <stdio.h>

int main() {

int num, i, flag = 0;
printf("Enter a positive integer except
0 and 1: ");
scanf("%d", &num);

// 0 and 1 are not prime numbers
// change flag to 1 for non-prime number
if (num == 0 || num == 1)
flag = 1;

for (i = 2; i <= num / 2; ++i) {

// if n is divisible by i, then n is not prime
// change flag to 1 for non-prime number
if (num % i == 0) {
flag = 1;
break;
}
}

// flag is 0 for prime numbers
if (flag == 0)
printf("%d is a prime number.", n);
else
printf("%d is not a prime number.", n);

return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus the c program To check whether the given string is Prime or not has been executed and the output is verified.

EX. No : 25 REPLACE ALL 0' S WITH 1' S

DATE :

AIM:

To replace all 0's with 1's.

ALGORITHM:

STEP 1: Take Input in num and initialize a variable num with with value 0

STEP 2: If num is equals to zero then update the value of num2 to 1

STEP 3: Iterate using a while loop until num is greater then 0

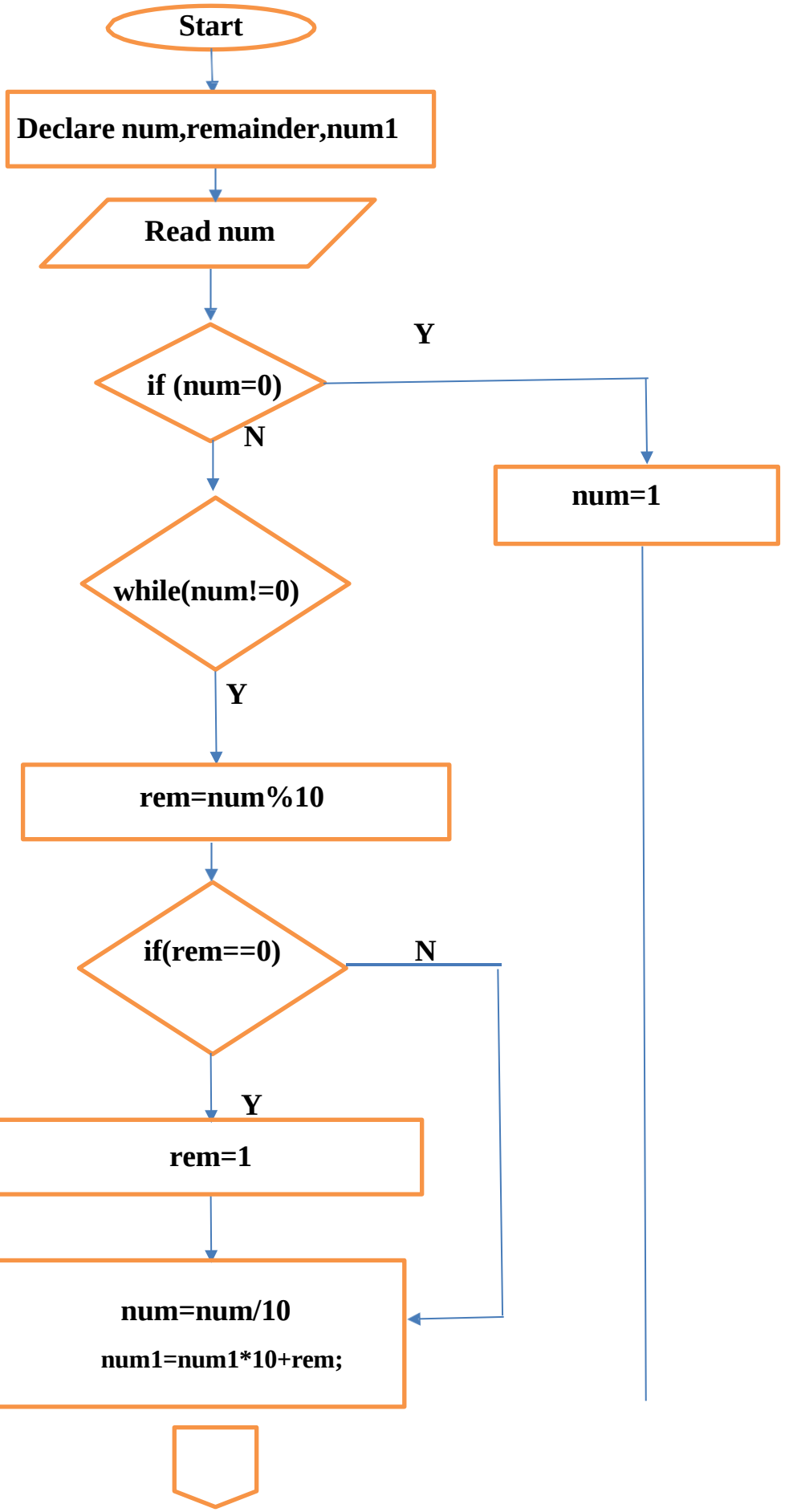
STEP 4: For each iteration initialize a variable rem and store unit digit of num

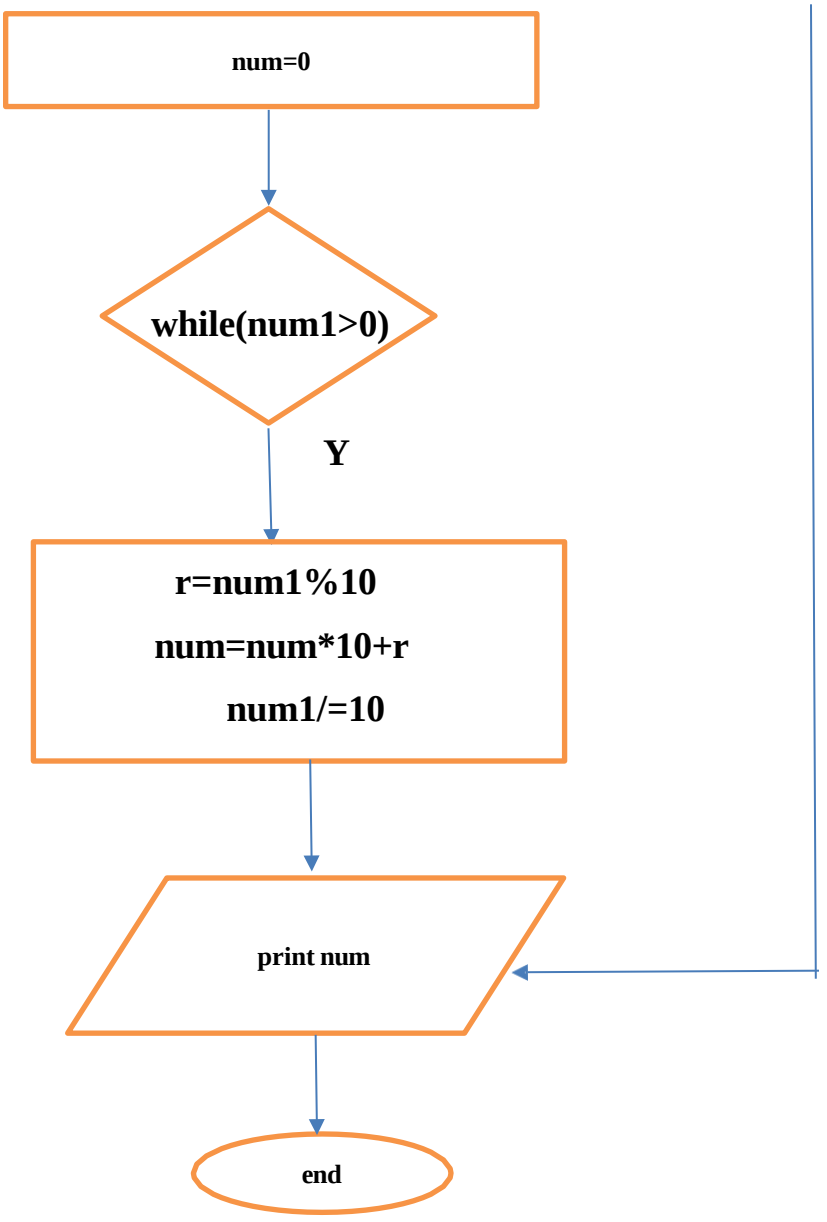
STEP 5: If rem equals to 0 then set rem to 1

STEP 6: Set num to num divide by 10 & num2 equals to num2*10+rem

STEP 7: Reverse and print num2

FLOW CHART:

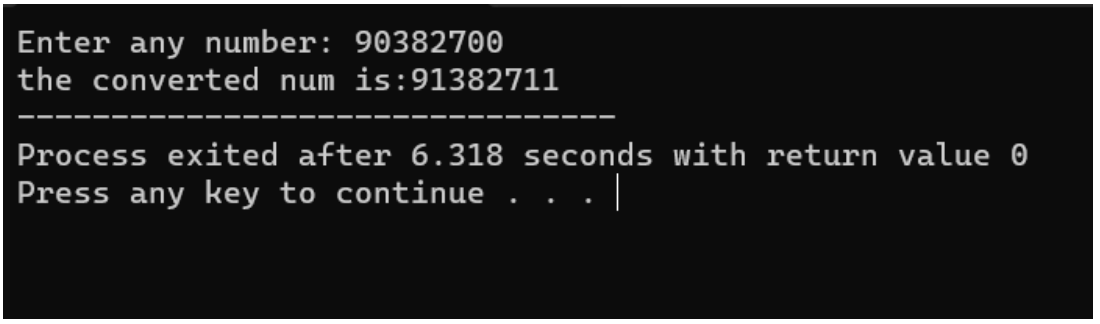

Start
Declare num,remainder,num1
Read num
if (num=0)
Y
N
num=1
while(num!=0)
Y
rem=num%10
if(rem==0)
N
Y
rem=1
num=num/10
num1=num1*10+rem;


num=0
while(num1>0)
Y
r=num1%10
num=num*10+r
num1/=10
print num
end

PROGRAM:

```
#include<stdio.h>

int main(){
int num,num1=0,rem,r;
printf("Enter any number: ");
scanf("%d", &num); if(num == 0)
{num1=1;}
//converting 0 to 1
else{
while(num>0){
rem = num%10;
if(rem == 0)
rem = 1;
num = num/10;
num1=num1*10+rem;
}

num = 0 ; // Store the reverse of num1
while(num1>0)
{
 r = num1%10;
num= num*10 + r;
num1 /= 10;
}
}
printf("the converted number is: %d" ,num);
return 0;
}
```

OUTPUT:



Criteria	Maximum Marks	Marks Obtained
Aim and Algorithm	**05**	
Program and Output	**15**	
Viva	**05**	
Total	**25**	

RESULT:

Thus, the c program To replace all 0's with 1's has been executed and the output is verified.