1.(a) TO FIND THE SUM OF INDIVIDUAL DIGITS OF A POSITIVE INTEGER.

Aim: To Write a C program to find the sum of individual digits of a positive integer.

Algorithm:

1. Read the number n
2. Initialize sum _ 0
3. while n > 0
4. d _ n%10
5. sum _ sum+d
6. n _ n/10
7. print sum.

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n, sum=0,d;
clrscr();
printf("Enter any integer:");
scanf("%d", &n);
while(n>0)
{
d=n%10;
sum=sum+d;
n=n/10;
}
Printf("sum of individual digits is %d",sum);
getch();
}
```

Output:

Enter any integer: 1234
Sum of individual digits is: 10

Result: The program is compiled, executed and the output is verified.

(b). GENERATE FIBONACCI SERIES OF THE FIRST N TERMS OF THE SEQUENCE.

Aim: To Write a C program to generate fibonacci series of the first n terms of the sequence.

Algorithm:

1. Read the number of terms n
2. Initialize a _ 0, b _ 1
3. print a and b values
4. for i _ 3 to n

a. increment the i value
b. c _ a+b
c. print c value
d. a _ b
e. b _ c

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=0,b=1,c,n,i;
clrscr();
printf("Enter no. of terms:");
scanf("%d", &n);
printf("The Fibonacci sequence is:");
printf("%d%d", a,b);
for(i=3;i<=n;i++)
{
c=a+b;
printf("%d",c);
a=b;
b=c;
}
getch();
}
```

Output:

```
Enter no of items: 5
The Fibonacci sequence is
0 1 1 2
```

Result: The program is compiled, executed and the output is verified.

(c). GENERATE ALL THE PRIME NUMBERS BETWEEN 1 AND N .

Aim: To Write a C program to generate all the prime numbers between 1 and n

.

Algorithm:

1. Read n value
2. Initialize count _ 0
3. for i _ 2 to n
a. for j _ 1 to i
b. if i mod j is equal to 0
c. then increment count
d. if count is equal to 2
e. then print i value.

Program:

```
#include<stdio.h>
#Include<conio.h>
void main()
{
int i, j, n, count=0;
clrscr();
printf("Enter the limit:");
scanf("%d", &n);
printf("The prime numbers are:");
for(i=2;i<=n;i++)
{
for(j=1;j<=i;j++)
{
if(i%j==0)
count++;
}
if(count==2)
printf("%d\t", i);
}
getch();
}
```

Output:

Enter the limit: 4
The prime numbers are:
2 3 5 7

Result: The program is compiled, executed and the output is verified.

(d). PROGRAM TO FIND THE ROOTS OF A QUADRATIC EQUATION.

Aim: To Write a C program to find the roots of a quadratic equation.

Algorithm:

1. Read a,b,c values
2. Initialize d _ b*b-4*a*c
3. if d==0
a. then print “roots are real and equal”
b. r1_ -b/2*a,r2 _ r1
4. else if d>0
a. then print “roots are real and distinct”
b. r1_ (-b+sqrt(d))/2*a, r2_ (-b-sqrt(d))/2*a
5. else if d<0
a. then print “roots are complex”
b. rp_ -b/2a, imp_ sqrt(d)/2*a
c. print r1 and r2.

Program:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
float a,b,c,d,r1,r2,imp,rp;
clrscr();
printf(“Enter a,b,c:”);
scanf(“%f%f%f”,&a,&b,&c);
d=b*b-4.0*a*c;
if(d= =0)
{
Printf(“roots are real and equal”);
r1=-b/2*a;
r2=r1;
printf(“root1=%f”,r1);
printf(“root2=%f”,r2);
}
else if(d>0)
{
Printf(“roots are real and unequal”);
```

```
r1=(-b+sqrt(d))/2*a;
r2=(-b-sqrt(d))/2*a;
printf("root1=%f",r1);
printf("root2=%f",r2);
}
else if(d<0)
{
d=-d;
printf("roots are complex");
rp=-b/2*a;
imp=sqrt(d)/2*a;
printf("root1=%f+i%f",rp,imp);
printf("root2=%f-i%f",rp,imp);
}
getch();
}
```

Output:

```
Enter a,b & c: 1 5 3
Roots are real & unequal
```

Result:

Thus the c program is executed and verified successfully.

(e). C PROGRAM TO PRINT PASCALS TRIANGLE

Aim: To write a c program to print pascals triangle

Algorithm:

1. Take the input from the user of the number of rows required in the pascal triangle.
2. Run the outer loop from 0 to n.
3. Run the inner loop from 0 to the outer loop.
4. Now for every pair of values of the inner loop and outer loop we will find the corresponding binomial coefficient.
5. To find the binomial coefficients we first need to find the combination and factorial.
6. Now for every point add the value of two points closest to it in the above row.
7. Repeat this operation and you will get your pascal triangle.

Program:

```
#include <stdio.h>

void main()
{
   int no_row,c=1,blk,i,j;
   printf("Input number of rows: ");
   scanf("%d",&no_row);
   for(i=0;i<no_row;i++)
   {
    for(blk=1;blk<=no_row-i;blk++)
      printf("  ");
      for(j=0;j<=i;j++)
      {
         if (j==0||i==0)
            c=1;
         else
            c=c*(i-j+1)/j;
         printf("% 4d",c);
      }
```

Output:

```
Input number of rows: 5
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
```

Result: The program is compiled, executed and the output is verified.

(f). FIND THE FACTORIAL OF A GIVEN INTEGER BY USING RECURSIVE AND NON-RECURSIVE FUNCTIONS.

Aim: To Write a C program to find the factorial of a given integer by using recursive and non-recursive functions.

i)Recursive Algorithm:

1. Define the recursive function
2. Read the number n
3. if n is equal to 0
4. then print “factorial of 0 is 1”
5. else call the recursive function
6. print the factorial value

Program:

```
#include<stdio.h>
#include<conio.h>
unsigned int factorial(int n);
void main()
{
int n,i;
long int fact;
clrscr();
printf("Enter the number: ");
scanf("%d",&n);
if(n==0)
printf("Factorial of 0 is 1\n");
else
printf("Factorial of %d Using Recursive Function is %d\n",n,factorial(n));
getch();
}
```

Output:

```
Enter number: 5
Factorial of 5 using recursive function is: 120
```

ii)Non-Recursive Algorithm: main program

Step 1: start
Step 2: read n
Step 3: call the sub program fact(n)
Step 4: print the f value
Step 5: stop

Sub program: fact

Step 1: initialize the f=1
Step 2: if n==0 or n=1 return 1 to main program. If not goto step 3
Step 3: perform the looping operation as follows
For i=1 i<=n; i++
Step 4: f=f*i
Step 5: return f value to the main program

Program:

```
#include<stdio.h>
#include<conio.h>
int fact(int n) //starting of the sub program
{
int f=1,i;
if((n==0)||(n==1))
return(1);
else
for(i=1;i<=n;i++)
return(f);
}
void main()
{
int n;
clrscr();
printf("enter the number :");
scanf("%d" ,&n);
printf("factorial of number %d", fact(n));
getch();
}
```

Output:

1.Enter a number: 7
Factorial of number: 5040

Result: The program is compiled, executed and the output is verified.

(g). FIND THE GCD(GREATEST COMMON DIVISOR) OF TWO GIVEN INTEGERS BY USING RECURSIVE FUNCTION

Aim: To Write a C program to find the GCD(greatest common divisor) of two given integers by using recursive function.

Algorithm:

1. Define the recursive function
2. Read the a,b values
3. Call the recursive function

a. if n>m
b. then return the function with parameters m,n
c. if n==0
d. then return m
e. else return the function with parameters n,m%n.

Program:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
unsigned int GCDRecursive(unsigned m, unsigned n);
int main(void)
{
int a,b;
clrscr();
printf("Enter the two numbers whose GCD is to be found: ");
scanf("%d%d",&a,&b);
printf("GCD of %d and %d Using Recursive Function is %d\n",a,b,GCDRecursive(a,b));
getch();
}
unsigned int GCDRecursive(unsigned m, unsigned n)
{
if(n>m)
return GCDRecursive(n,m);
if(n==0)
return m;
else
return GCDRecursive(n,m%n);
}
```

Output:

```
Enter the two numbers whose GCD is to be found 18     6
GCD of 18 and 6 Using Recursive Function is 6
```

Result: The program is compiled, executed and the output is verified

2.(a). FIND BOTH THE LARGEST AND SMALLEST NUMBER IN LIST OF INTEGERS

Aim: - To write a C program to find both the largest and smallest number in list of integers.

Algorithm:

1. Start
2. Read n
3. for i _ 0 to n
4. do read a[i]
5. small _ a[0]
6. for i _ 0 to n
7. do if small > a[i]
8. then small _ a[i]
9. write small
10. large _ 0
11. for i _ 0 to n
12. do if large <a[i]
13. then large _ a[i]
14. write large
15. Stop

Program:

```
#include <stdio.h>
#include <conio.h>
Void main()
{
int i,n,small=0,large=0;
int a[30];
clrscr();
printf("\n Enter size of the array:");
scanf("%d",&n);
printf("\n Enter values in array elements:");
for(i=0;i<n;i++)
{
scanf("%d" ,&a[i]);
}
small = a[0];
for(i=0;i<n;i++)
{
```

```
if(small > a[i])
small = a[i];
}
printf("\n The smallest element in given array is %d" ,small);
large=0;
for(i=0;i<n;i++)
{
if(large < a[i])
large = a[i];
}
printf("\n The largest element in given array is %d",large);
printf("\n :End of the Main Program:");
getch();
}
```

Output:

Input :
Enter size of the array: 9
Enter values in array elements:
96 46 86 6 36 76 26 16 56
Output:
The smallest element in given array is 6
The largest element in given array is 96

Result: The program is compiled, executed and the output is verified.

(b). ADDITION ON TWO MATRICES

Aim: To Write a c- program that to perform addition on two matrices.

Algorithm:

1. Start
2. read r1,r2,c1,c2
3. if r1 \ r2 and c1 \ c2
4. then “matrix addition is not possible”
5. else
6. do init_mat(a,r1,c1)
7. print_mat(a,r1,c1)
8. init_mat(b,r2,c2)
9. print_mat(b,r2,2)
10. add_mat(a,b,c,r1,c1)
11. print_mat(c,r1,c1)
12. Stop

Program:

```
#include <stdio.h>
#include<conio.h>
void main()
{
int m, n, c, d, first[10][10], second[10][10], sum[10][10];
printf("Enter the number of rows and columns of matrix\n");
 scanf("%d%d", &m, &n);
printf("Enter the elements of first matrix\n");
for (c = 0; c < m; c++) for (d = 0; d < n; d++)
scanf("%d", &first[c][d]);
printf("Enter the elements of second matrix\n");
for (c = 0; c < m; c++)
for (d = 0 ; d < n; d++)
scanf("%d", &second[c][d]) ;
printf("Sum of entered matrices:-\n");
for (c = 0 ; c < m; c++)
{
for (d = 0 ; d < n; d++)
 {
sum[c][d] = first[c][d] + second[c][d];
printf("%d\t", sum[c][d]);
}
printf("\n");
}
```

```
getch ( ) ;
}
```

Output :

```
Enter the number of rows and columns of matrix 3  2
 Enter the elements of first matrix   5       8
                                      4       1
                                      5       2
Enter the elements of second matrix   2     6
                                        4    7
                                        1     2
Sum of entered matrices               7    14
                                      8     8
                                      6     4
```

Result: The program is compiled, executed and the output is verified.

(c). MATRIX MULTIPLICATION ON TWO MATRICES.

Aim: Write A C- Program That To Perform Matrix Multiplication On Two Matrices.

Algorithm:

Step 1: start
Step 2: Read rows and columns of two matrices
Step 3: Read elements of first and second matrix to arrays A[ ][ ] and B[ ][ ]
Step 4: if columns [A] ≠ rows [B]
Then error "incompatible dimensions – Multiplication Not possible"
Repeat step 2 if required.
Step 5: For i varies from 1 to rows [A]
Step 6: For j varies from 1 to columns [B]
Step 7: initialize C[i,j]=0
Step 8: For k varies from 1 to columns [A]
Step 9: C[i, j]=C[i, j]+A[i, k]*B[k, j] , Repeat 5to 9 till the end of loops.
Step 10:Print result as C[ ][ ]
Step 11: Stop

Program:

```
#include <stdio.h>
int main()
{
int m, n, p, q, i, j, k, sum = 0;
int first[10][10], second[10][10], multiply[10][10];
printf("Enter the number of rows and columns of first matrix\n");
scanf("%d%d", &m, &n);
printf("Enter the number of rows and columns of second matrix\n");
scanf("%d%d", &p, &q);
if (n != p)
printf("Matrices with entered orders can't be multiplied with each other.\n");
else
{ printf("Enter the elements of first matrix\n");
for (i = 0; i < m; i++)
for (j = 0; j < n; j++)
scanf("%d", &first[i][j]);
printf("Enter the elements of second matrix\n");
for (i = 0; i < p; i++)
for (j = 0; j < q; j++)
scanf("%d", &second[i][j]);
for (i = 0; i < m; i++) {
for (j = 0; j < q; j++) {
multiply[i][j]=0;
for (k = 0; k < p; k++) {
multiply[i][j] = multiply[i][j] + first[i][k]*second[k][j];
```

```
}
}
}
printf("Product of entered matrices:-\n");
for (i = 0; i < m; i++) {
for (j = 0; j < q; j++)
printf("%d\t", multiply[i][j]);
printf("\n");
}
}
return 0;
}
```

Output:

```
Enter the number of rows and columns of first matrix 3  3
Enter the number of rows and columns of second matrix 3   3
Enter the elements of first matrix
1    2   4
5    2    1
 4   5    2
the elements of second matrix
1     2   4
5     2   1
 4    5    2
Product of entered matrices
10  18   28
50  18    7
40  45   14
```

Result: The program is compiled, executed and the output is verified.

3.(a).TO INSERT A SUB-STRING IN TO A GIVEN MAIN STRING FROM A GIVEN POSITION.

Aim: - To Write A C- Program That Uses Functions To Insert A Sub-String In To A Given Main String From A Given Position.

Algorithm:

1. start
2. read str (string)
3. read n(position), substr (sub string)
4. ins_substr(str,substr,p,n)
5. stop

Program:

```
#include <stdio.h>
#include <conio.h>
void ins_substr(char [], char [], int, int);
main()
{
int p,n,i,j;
char str[50],substr[50];
clrscr();
puts("\n Enter the String:");
gets(str);
fflush(stdin);
printf("Enter the specific position ");
scanf("%d",&p);
printf("\n Enter the Number of Characters:");
scanf("%d",&n);
fflush(stdin);
puts("\n Enter Sub-String:");
gets(substr);
ins_substr(str,substr,p,n);
printf("\n :: End of the main program ::");
getch();
}
void ins_substr(char str[], char substr[], int p, int n)
{
int q,i,j;
```

```
q=p-1;
for(i=q,j=n;str[i]!='\0';i++,j++)
substr[j]=str[i];
substr[j]='\0';
for(j=0,i=q;substr[j]!='\0';j++,i++)
str[i]=substr[j];
str[i]='\0';
printf("\n The string after inserting substring :");
puts(str);
}
```

Output:

Case - 1

Input :

Enter the String: HELO WORLD
Enter the specific position : 3
Enter the Number of Characters: 1
Enter Sub-String: L

Output :

The string after inserting substring : HELLO WORLD
:: End of the main program ::

Case - 2

Input :

Enter the String: HELLO
Enter the specific position : 5
Enter the Number of Characters: 5
Enter Sub-String: WORLD

Output :

The string after inserting substring : HELLO WORLD
:: End of the main program ::

Result: The program is compiled, executed and the output is verified.

(b). FUNCTIONS TO DELETE N – CHARACTRES FROM A GIVEN POSITION IN A GIVEN STRING.

Aim: - Write A C- Program That Uses Functions To Delete N – Characters From A Given Position In A Given String.

Algorithm:

1. start
2. read str(main string)
3. read p (position)
4. read n (number of characters to delete)
5. del_str(str,p,n)
6. stop

del_str(str,p,n)

1. for i _ 0 , j _ 0 to Length[str]
2. do if i = p-1
3. i _ i + n
4. str[j] _ str[i]
5. str[j] _ NULL
6. print str

Program:

```
#include <stdio.h>
#include <conio.h>
void del_str(char [],int, int);
main()
{
int n,p;
char str[30];
clrscr();
printf("\n Enter the String::");
gets(str);
fflush(stdin);
printf("\n Enter the position from where the characters are to be deleted:");
scanf("%d",&p);
printf("\n Enter Number of characters to be deleted:");
scanf("%d",&n);
/* function call to deletion of n-characters */
del_str(str,p,n);
printf("::End of the Main program::");
```

```
getch();
}
void del_str(char str[],int p, int n)
{
int i,j;
for(i=0,j=0;str[i]!='\0';i++,j++)
{
if(i==(p-1))
{
i=i+n;
}
str[j]=str[i];
}
str[j]='\0';
/* the string after deletion */
puts(" The string after deletion of characters::");
puts(str);
}
```

Output:

Case - 1

Input :

Enter the String: ABCD EFGH IJKL
Enter the position from where the characters are to be deleted: 5
Enter Number of characters to be deleted: 4

Output :

The string after deletion of characters:: ABCD IJKL
:: End of the main program ::

Result: The program is compiled, executed and the output is verified.

(c). TO DETERMINE IF THE GIVEN STRING IS A PALINDROME OR NOT

Aim: Write a C- Program To Determine If The Given String Is A Palindrome Or Not.

Algorithm:

1. Start
2. read str (string)
3. len _ Length[str]
4. for i _ 0 (increment step), j _ len-1 (decrement step) to Length[str]
5. do str[i] \ str[j]
6. print " not palindrome"
7. stop
8. print "palindrome"
9. stop

Program:

```
#include <stdio.h>
#include <conio.h>
main()
{
int i,n,j,len=0;
char str[30];
clrscr();
printf("\n Enter String:");
gets(str);
for(i=0;str[i]!='\0';i++)
len++;
printf("\n The length of the string is %d",len);
for(i=0,j=len-1;str[i]!='\0';i++,j--)
{
if(str[i]!=str[j])
{
printf("\n :The given string is not a palindrome:");
getch();
exit(0);
} }
printf("\n :the given string is palindrome:");
getch();
}
```

Output:

Case - 1

Input :

Enter the String: MALAYALAM
The length of the string is 9

Output :

:the given string is palindrome:

Result: The program is compiled, executed and the output is verified.

(d). TO COUNT THE LINES, WORDS, CHARACTERS IN A GIVEN TEXT

Aim: - Write A C- Program To Count The Lines, Words, Characters In A Given Text.

Algorithm:

1. start
2. read text
3. while text[i] != EOF
4. do i _ i+1
5. print i
6. for i _ 0 to Length[text]
7. do ch++
8. if text[i] = 32 and text[i+1] \ ‘ ‘
9. then w++
10. sp++
11. if text[i] = ‘\n’
12. then l++
13. w++
14. print ch
15. print w+1
16. print l
17. print sp

Program:

```
#include <conio.h>
#include <stdio.h>
main()
{
char text[200];
int i,l,ch,w,sp;
clrscr();
i=0;
printf("\n Enter lines of text and press ^Z");
while((text[i]=getchar())!=EOF)
{
i++;
}
printf("\n The number of characters is %d",i);
text[i]='\0';
l=0;
ch=w=sp=0;
```

```
for(i=0;text[i]!='\0';i++)
{
ch++;
if(text[i]==32 && text[i+1] != ' ')
{
w++;
sp++;
}
if(text[i] == '\n')
{
l++;
w++;
}
}
printf("\n Total size of the text : %d",ch);
printf("\n Number of Words : %d",w+1);
printf("\n Number of Lines : %d",l);
printf("\n Number of Spaces : %d",sp);
getch();
}
```

Output:

```
Enter lines of text and press ^Z”” ABCD EFGH IJKL MNOP
Total size of the text: 14
Number of Words:8
Number of Lines:1
Number of Spaces:4
```

Result: The program is compiled, executed and the output is verified.

(e). PROGRAM TO CONSTRUCT A PYRAMID OF NUMBERS.

Aim: Write an algorithm, flowchart and program to construct a pyramid of numbers.

Algorithm:

Step 1: Start
Step 2: Read height n of the pyramid
Step 3: for j := 0 to n do
Step 3.1: for k:= 1 to 2*(n-j) do
Step 3.1.1: Print blank space
Step 3.1.2: k := k+1
Step 3.2: for i:= -j to j do
Step 3.2.1: print abs(i)
Step 3.2.2: i := i+1
Step 3.3: j := j+1
Step 4: Stop

Program:

```
#include <stdio.h>
#include <conio.h>
main()
{
int i,j,k,n;
clrscr();
printf("Enter the height of the pyramid");
scanf("%d",&n);
for(j=0;j<=n;j++)
{
for(k=1;k<=2*(n-j);k++)
printf(" ");
for(i=-j;i<=j;i++)
printf("%d ",abs(i));
printf("\n");
}
}
```

Output:

Enter the height of the pyramid: 2

1
2 2

Result: The program is compiled, executed and the output is verified.

4.(a). C PROGRAM WHICH COPIES ONE FILE TO ANOTHER.

Aim: Write a C program which copies one file to another.

Algorithm:

1: Declare the variables and file pointer
2: Check the number of arguments in command line
if arg is not equal to 3
then print “invalid arguments”
exit
3: Open an existed file in read mode
4: if file not found
then print “file can not be open”
5: Assign 2nd argument in command line to a variable
6: Read the contents from existed file and reverse first n characters in the string from file.

Program:

```
#include <stdio.h>
#include <conio.h>
#include <process.h>
void main(int argc, char *argv[])
{
FILE *fs,*ft;
char ch;
clrscr();
if(argc!=3)
{
puts("Invalid number of arguments.");
exit(0);
}
fs = fopen(argv[1],"r");
if(fs==NULL)
{
puts("Source file cannot be opened.");
exit(0);
}
ft = fopen(argv[2],"w");
if (ft==NULL)
{
puts("Target file cannot be opened.");
fclose(fs);
```

```
exit(0);
}
while(1)
{
ch=fgetc(fs);
if (ch==EOF)
break;
else
fputc(ch,ft);
}
fclose(fs);
fclose(ft);
getch();
}
```

Output:

File created is passed as parameter:
File is copied

Result: The program is compiled, executed and the output is verified.

(b). C PROGRAM TO REVERSE THE FIRST N CHARACTERS IN A FILE.

Aim: Write a C program to reverse the first n characters in a file.

Algorithm:

1: Declare the variables and file pointer
2: Check the number of arguments in command line
if arg is not equal to 3
then print “invalid arguments”
exit
3: Open an existed file in read mode
4: if file not found then print “file can not be open”
5: Assign 2nd argument in command line to a variable
6: Read the contents from existed file and reverse first n characters in the string from file.

Program:

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <process.h>
void main(int argc, char *argv[])
{
char a[15];
char s[20];
char n;
int k;
int j=0;
int i;
int len;
FILE *fp;
if(argc!=3)
{
puts("Improper number of arguments.");
exit(0);
}
fp = fopen(argv[1],"r");
if(fp == NULL)
```

```
{
puts("File cannot be opened.");
exit(0);
}
k=atoi(argv[2]);
n = fread(a,1,k,fp);
a[n]='\0';
len=strlen(a);
for(i=len-1;i>=0;i--)
{
s[j]=a[i];
printf("%c",s[j]);
j=j+1;
}
s[j+1]='\0';
getch();
}
```

Output:

Abc.txt: He is a good boy
Output: yob doog a si eH

Result: The program is compiled, executed and the output is verified.

5.(a). PROGRAM TO IMPLEMENT THE LINEAR SEARCHING METHOD.

Aim: Write a C program to Implement the following searching method.

Algorithm:

1. Step 1: set pos = -1
2. Step 2: set i = 1
3. Step 3: repeat step 4 while i <= n
4. Step 4: if a[i] == val
5. set pos = i
6. print pos
7. go to step 6
8. [end of if]
9. set ii = i + 1
10. [end of loop]
11. Step 5: if pos = -1
12. print "value is not present in the array "
13. [end of if]
14. Step 6: exit

Program:

```
#include<stdio.h>
main()
{
int a[10],i,n,key,co=0;
clrscr();
printf("how many you want");
scanf("%d",&n);
printf("enter array elements:");
for(i=0;i<n;i++)
{
scanf("%d",&a[i]);
}
printf("enter the searching elements");
scanf("%d",&key);
search(a,n);
}
Void search(int a[10], int n)
{
int i;
for(i=0;i<n;i++)
```

```
{
if(a[i]==key)
co++;
}
if(co>0)
printf("Element is found");
else
printf("Not found");
getch();
}
```

Output:

```
how many you want5
enter array elements:3 1 7 12 45
enter the searching elements12
Element is found
```

Result: The program is compiled, executed and the output is verified.

5.(b) C PROGRAM TO IMPLEMENT THE BINARY SEARCHING METHOD.

Aim: Write a C program to Implement the following searching method.

Algorithm:

1. Read the search element from the user.
2. Find the middle element in the sorted array.
3. Compare the search element with the middle element in the sorted array.
4. If both are matched, then display "Given element is found!!!" and terminate the function.
5. If both are not matched, then check whether the search element is smaller or larger than the middle element.
6. If the search element is smaller than the middle element, repeat steps 2, 3, 4 and 5 for the left subarray of the middle element.
7. If the search element is larger than the middle element, repeat steps 2, 3, 4 and 5 for the right subarray of the middle element.
8. Repeat the same process until we find the search element in the array or until the subarray contains only one element.
9. If that element also doesn't match with the search element, then display "Element is not found in the array!!!" and terminate the function**.**

Program :

```
#include<stdio.h>
main()
{
int a[10],i,j,t,n,key,low,high,co;
clrscr();
printf("how many you want");
scanf("%d",&n);
printf("enter array elements:");
for(i=0;i<n;i++)
{
scanf("%d",&a[i]);
}
for(i=0;i<n;i++)
{
for(j=0;j<n-i-1;j++)
{
if(a[j]>a[j+1])
{
```

```
t=a[j];
a[j]=a[j+1];
a[j+1]=t;
} } }
low=0;
high=n-1;
printf("enter the searching elements");
scanf("%d",&key);
co=binarysearch(a,low,high,key);
if(co==-1)
printf("Not found");
else
printf("Element is found");
getch();
}
Rbinarysearch(int a[10],int low,int high,int key)
{
int mid;
if(low>high)
return(-1);
mid=(low+high)/2;
if(key==a[mid])
return(mid);
if(key<a[mid])
return(Rbinarysearch(a,low,mid-1,key));
else
return(Rbinarysearch(a,mid+1,high,key));
}
```

Output:

```
how many you want  5
enter array elements:32 1 45 67 98
enter the searching elements98
Element is found
```

Result: The program is compiled, executed and the output is verified.

(c). PRINT THE STUDENT DETAILS AND PRINT THEM

Aim: To Write a program to take input of name, rollno and marks obtained by a student in 4 subjects of 100 marks each and display the name, rollno with percentage score secured.

Algorithm:

1. Start
2. Define variables: name, rollno, sub1, sub2, sub3, sub4, sum, score
3. Take input from keyboard for all the input variables
4. Calculate the sum of marks of 4 subjects and also calculate the percentage score as:
sum = sub1 + sub2 + sub3 + sub4;
score = (sum/400) * 100
5. Display the name, roll number and percentage score.
6. Stop

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
char name[20];
int rollno;
float sub1, sub2, sub3, sub4, , sum, score;
printf("Enter name of student: ");
scanf("%s",&name[]);
printf ("\n Enter Roll Number: ");
scanf("%d", &rollno);
printf ("\n Enter Marks in 4 Subjects:\n");
scanf("%f%f%f%f", &sub1, &sub2, &sub3, &sub4);
sum=sub1+sub2+sub3+sub4;
score = (sum/500)*100;
printf("\n Name of student: %s", name[]);
printf("\n Roll Number: %d", rollno);
printf ("\nPercentage score secured: %2.2f%c", score,'%');
getch();
}
```

Output:

```
Enter name of student: Ajit Singh
Roll Number: 25
Enter Marks in 4 Subjects:
50
75
85
62
Name of student: Ajit Singh
Roll Number: 25
Percentage score secured: 68.00%
```

Result: The program is compiled, executed and the output is verified.

6.(a). PRINT WHETHER A GIVEN NUMBER IS EVEN OR ODD.

Aim: To Write a program to print whether a given number is even or odd.

Algorithm:

1: START
2:Receive data from a user or dynamically assign value.
3: if num percent 2 == 0. Even Integer should be printed.
4: Else, Print an Odd Number
5: Print an Odd Number
6: END

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int num;
printf("Enter the number: ");
scanf("%d",&num);
if(num%2==0)
printf("\n %d is even", num);
else
printf("\n %d is odd", num);
getch();
}
```

Output:

```
Enter the number: 6
6 is even
```

Result: The program is compiled, executed and the output is verified.

(b). TO FIND WHETHER A CHARACTER IS CONSONANT OR VOWEL USING SWITCH STATEMENT

Aim: To Write a program to find whether a character is consonant or vowel using switch statement.

Algorithm:

1. Input an alphabet from user. ...
2. Switch the value of ch .
3. For ch , there are 10 possibilities for vowel we need to check i.e. a , e , i , o , u , A , E , I , O and U .
4. Write all 10 possible cases for vowels and print “Vowel” for each case .

Program:

```
#include <stdio.h>
void main()
{
char ch;
printf(“Enter any alphabet:”);
scanf(“%c”, &ch);
switch(ch)
{
case ,,a‟‟:
case ,,A‟‟:
printf(“Vowel”);
break;
case ,,e‟‟:
case ,,E‟‟:
printf(“Vowel”);
break;
case ,,i‟‟:
case ,,I‟‟:
printf(“Vowel”);
break;
case ,,o‟‟:
case ,,O‟‟:
printf(“Vowel”);
break;
case ,,u‟‟:
case ,,U‟‟:
```

```
printf(“Vowel”);
break;
default:
printf(“Consonant”);
}
}
```

Output:

```
Enter an Alphabet
e
e is VOWEL
Enter an Alphabet
Z
Z is CONSONANT
```

Result: The program is compiled, executed and the output is verified.

7. PROGRAM TO DISPLAY THE FOLLOWING PATTERN.

```
*
* *
* * *
* * * *
* * * * *
```

Aim: To Write a program to display the following pattern.

Algorithm:

- Print the star pattern from the end of each Row.
- Complete the last column of each Row.
- Start from the Second Last Column of the second row.
- Repeat till the number of rows is specified by the User.

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
for(i=1; i<=5;i++)
{
for(j=1;j<=i;j++)
{
printf("*");
}
printf("\n");
}
getch();
}
```

Output:

```
*
* *
* * *
* * * *
* * * * *
```

Result: The program is compiled, executed and the output is verified.

8. TO FIND BIGGEST AMONG THREE NUMBERS USING POINTER.

Aim: Write a program to find biggest among three numbers using pointer.

Algorithm:

1:Initialize three integer variables as num1, num2, and num3 to store three input numbers.
2. Read three integer numbers num1, num2, and num3 from the user.
3. Compare num1 with num2 and num3 to find the largest of num1, num2, and num3.

Program:

```
#include<stdio.h>
#include<conio.h>
int main()
{
int a,b,c;
int*ptra=&a,*ptrb=&b,*ptrc=&c;
printf("enter three values");
scanf("%d%d%d",ptra,ptrb,ptrc);
printf("a=%d\n b=%d\n c=%d\n",*ptra,*ptrb,*ptrc);
if((*ptra>*ptrb && *ptra>*ptrc))
printf("biggest number=%d",*ptra);
else if((*ptrb>*ptra && *ptrb>*ptrc))
printf("biggest number =%d",*ptrb);
else
printf("biggest number=%d",*ptrc);
getch();
return 0;
}
```

Output:

Enter three numbers:

10 14 6

The biggest number is 14

Result: The program is compiled, executed and the output is verified.

9. PERFORM THE ARITHMETIC EXPRESSION USING SWITCH STATEMENT

Aim: Write a Program to perform the arithmetic expression using switch statement.

Algorithm:

1: start
2: input a,b,op
3: switch(op)
4: case '+': print 'sum of a& b is' ,a+b
5: case '-': print 'difference of a& b is' ,a-b
6: case '*': print 'product of a& b is', a*b
7: case '/': print 'quotient of a& b is', a/b
8: default: invalid option
9: stop

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
int op;
clrscr();
printf(" 1.addition\n 2.subtraction\n 3.multiplication\n 4.division\n");
printf("enter the values of a & b");
scanf("%d%d",&a,&b);
printf("enter your choice : ");
scanf("%d",&op);
switch(op)
{
case 1 :printf("sum of %d and %d=%d",a,b,a+b);
break;
case 2 :printf("difference of %d and %d=%d",a,b,a-b);
break;
case 3 :printf("multiplication of %d and %d=%d",a,b,a*b);
break;
case 4 :printf("Division of two numbers is %d=",a/b);
break;
default : printf(" Enter Your Correct Choice.");
}
getch();
}
```

Output:

1. Addition
2. Subtraction
3. Multiplication
4. Division

Enter your choice : 1
Enter a and b values 10 20
Sum of 10 and 20 = 30

Result: The program is compiled, executed and the output is verified.

10. WRITE A PROGRAM TO FIND THE FACTORIAL OF A GIVEN NUMBER

Aim: Program to find the factorial of a given number

Algorithm:

Step1: start
Step2: input n,i,f
Step3: Initialize f=i=1
Step4: if(i<=n) repeat steps 5 and 6 otherwise go to step 7
Step5: f=f*i
Step6: i=i+1
Step7: print f
Step8: stop

Program:

```
void main()
{
int n,i,f;
f=i=1;
clrscr();
printf("enter a number");
scanf("%d",&n);
while(i<=n)
{
f*=i;
i++;
}
printf("the factorial of %d is %d",n,f);
getch();
}
```

Output:

Enter a number 5
The factorial of 5 is 120

Result: The program is compiled, executed and the output is verified.

11. PROGRAM TO FIND FACTORIAL OF A GIVEN NUMBER USING FUNCTION.

Aim: Program to find factorial of a given number using function.

Algorithm:

step 1. Start
step 2. Read the number n
step 3.call function f=fact(n)
step 4.print f
step 5. Stop

Program:

```
#include<stdio.h>
#include<math.h>
void main()
{
int n;
long int f;
long int fact(int);
clrscr();
printf("enter a number");
scanf(“%d”,&n);
f=fact(n);
printf("\nfactorial of a given no is: %d ",f);
getch();
}
long int fact(int n)
{
int i;
long int f=1;
for(i=1;i<=n;i++)
{
f=f*i;
}
return f;
}
```

Output:

Enter a number 5
Factorial of a given no is: 120

Result: The program is compiled, executed and the output is verified.

12. PROGRAM TO FIND TOTAL MARK OF N STUDENTS USING STRUCTURE

Aim: To write c Program to find total mark out of 5 subjects of n students.

Algorithm:

1: start
2: Define user data type for student.
3. Read number of students n
4: for i varies from 0 to n-1 , repeat steps 5 to 9 otherwise goto step 10
5: Read student struct variable, s[i].name,s[i].rollno
6: for j varies from 0 to 4, repeat steps 7,8 otherwise go to step9
7: read mark as s[i].mark[j]
8 :s[i].total = s[i].total+s[i].mark[j]
9:display s[i].name,s[i].rollno, and s[i].total
10:stop

Program:

```
#include<stdio.h>
struct student
{
char name[10];
int rollno;
int subject[5],total;
};
main ( )
{
static struct student s[100];
int n,i,j;
clrscr()

printf("enter the no.of students");
scanf("%d",&n);
printf("enter the marks of five subjects");
for(i=0;i<n;i++)
{
printf(“enter student roll number \n”);
scanf(“%d”,&s[i].rollno);
printf(“enter student name \n”);
scanf(“%s”, s[i].name);
printf("enter s[%d] student marks\n",i);
s[i].total=0;
for(j=0;j<5;j++)
{
scanf("%d",&s[i].subject[j]);
```

```
s[i].total=s[i].total+s[i].subject[j];
}
printf("%d\t%s\t%d\n",s[i].rollno,s[i].name,s[i].total);
}
}
```

Output:

```
enter the no.of students2
enter student roll number
1
enter student name
anu
enter the marks of five subjects
enter s[0] student marks
1
2
3
4
5
1 anu 15
enter student roll number

2
enter student name
akhil
enter s[1] student marks
12
32
14
15
65
2 akhil 138
```

Result: The program is compiled, executed and the output is verified.

13. PROGRAM TO PRINT THE ELEMENTS OF ARRAY USING POINTERS

Aim: Program to print the elements of array using pointers

Algorithm:

1: start
2: Read array a[] with n elements
3: initialize pointer p=&a[0] [or p=a]
4:if i<n go to next step otherwise go to step 7
5: print *(p+i)
6:i=i+1 go to step 4
7: stop

Program:

```
#include<stdio.h>
main()
{
int a[5]={5,4,6,8,9};
int *p=&a[0];
int i;
clrscr();
for(i=0;i<5;i++)
printf("%d ",*(p+i));
getch();
}
```

Output:

5 4 6 8 9

Result: The program is compiled, executed and the output is verified

14. PROGRAM TO CONCATENATE TWO STRINGS WITHOUT USING LIBRARY FUNCTIONS

Aim: To write c Program to print the concatenated string from two strings given.

Algorithm:

step1: start
step2: Read two strings in two arrays str1,str2; let 1=j=0;
Step 3:if str1[i] not equal to '\0' ,increment i, repeat this step
Step 4:if str2[j] not equal to '\0' go to next step otherwise go to step 7
Step 5:str1 [i]=str2[j]
Step 6: increment i and j ,go to step 4
Step 7:str1 [i]='\0'
Step 8: Print Concatenated string str1
step: stop

Program:

```
#include<stdio.h>
void main(void)
{
char str1[25],str2[25];
int i=0,j=0;
printf("\nEnter First String:");
gets(str1);
printf("\nEnter Second String:");
gets(str2);
while(str1[i]!='\0')
i++;
while(str2[j]!='\0')
{
str1[i]=str2[j];
j++;
i++;
}
str1[i]='\0';
printf("\nConcatenated String is %s",str1);
}
```

Output:

```
Enter the first string hello
Enter the second string world
Concatenated String is hello world
```

Result: The program is compiled, executed and the output is verified

15. WRITE A PROGRAM TO PERFORM THE ARITHMETIC EXPRESSION USING SWITCH STATEMENT

Aim: To write a c Program to perform the arithmetic expression using switch statement

Algorithm:

1: start
2: input a,b,op
3: switch(op)
4: case '+': print 'sum of a& b is' ,a+b
5: case '-': print 'difference of a& b is' ,a-b
6: case '*': print 'product of a& b is', a*b
7: case '/': print 'quotient of a& b is', a/b
8: default: invalid option
9: stop

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
int op;
clrscr();
printf(" 1.addition\n 2.subtraction\n 3.multiplication\n 4.division\n");
printf("enter the values of a & b");
scanf("%d%d",&a,&b);
printf("enter your choice : ");
scanf("%d",&op);
switch(op)
{
case 1 :printf("sum of %d and %d=%d",a,b,a+b);
break;
case 2 :printf("difference of %d and %d=%d",a,b,a-b);
break;
case 3 :printf("multiplication of %d and %d=%d",a,b,a*b);
break;
case 4 :printf("Division of two numbers is %d=",a/b);
break;
default : printf(" Enter Your Correct Choice.");
}
getch();
}
```

Output:

1. Addition
2. Subtraction
3. Multiplication
4. Division

Enter your choice : 1
Enter a and b values 10 20
Sum of 10 and 20 = 30

Result: The program is compiled, executed and the output is verified.

16. PROGRAM TO PRINT THE SUM OF 1 + 3 + 5 + 7 +……..+ N

Aim:

To write a c program to Print the sum of 1 + 3 + 5 + 7 +……..+ n

Algorithm:

```
#include <stdio.h>
#include<conio .h >
 void main ( )
{
int n, i, sum=0 ;
clrscr ( ) ;
printf("Enter the range = ");

scanf("%d",&n);
for(i =1; i <=n ; i = i + 2)
{
sum = sum + i ;
}
printf("Sum is = %d",sum);
getch ( ) ;
}
```

Output :

```
 Enter the range =9
Sum is = 25
```

Result: The program is compiled, executed and the output is verified.

17. TO ACCEPT 10 ELEMENTS AND SORT THEM IN DESCENDING ORDER USING ONE DIMENSIONAL ARRAY.

Aim: To write a c program to accept 10 elements and sort them in descending order using one dimensional array.

Algorithm:

1. Loop through the array and select an element.
2. Inner loop will be used to compare selected element from outer loop with rest of the elements of array.
3. If any element is greater than the selected element then swap the values.
4. Continue this process till entire list is sorted in descending order.

Program:

```
#include<stdio.h>
 #include<conio.h>
void main ( )
{
int i, j,temp,a[10];
 clrscr ( );
printf("Enter 10 integer numbers: \n");
for(i=0;i<10;i++);
scanf("%d",&a[i]);
for (i=0;i<10;i++)
{
for(j=i+1;j<10;j++)
{
if ( a[i] < a[j] )
{
temp=a[j]; a[ j]=a[i];
a [ i]=temp;
}
}
}
printf("\n\nThe 10 numbers sorted in descending order are: \n");
for(i=0;i<10;i++)
printf("%d\t",a[i]);
getch ( );
}
```

Output:

Enter 10 integer numbers: 5 2 10 7 6 1 4 3 8 9
The 10 numbers sorted in descending order are: 10 9 8 7 6 5 4 3 2 1

Result: The program is compiled, executed and the output is verified.

18. TO ACCEPT 10 NUMBERS AND MAKE THE AVERAGE OF THE NUMBERS USING ONE DIMENSIONAL ARRAY.

Aim:. To accept 10 numbers and make the average of the numbers using one dimensional array.

Algorithm:

1. We shall use a loop and sum up all values of the array.
2. Then we shall divide the sum with the number of elements in the array,
3. this shall produce average of all values of the array.

Program:

```
#include <stdio.h>
#include<conio .h >
 void main ( )
{
float a[15], sum =0 , avg , i ;
clrscr ( ) ;
printf("Enter values of 10 numbers = ");
for( i=0 ; i< 10 ; i ++)
{
scanf("%f ",& a [ i ] );

sum = sum + a[ i ] ;
}
avg = sum / 10 ;
printf("Average is = %f ",avg);
getch ( ) ;
}
```

Output:

Enter values of 10 numbers = 8 2 6 3 9 7 11 21 30 22
Average is = 11 . 900000

Result: The program is compiled, executed and the output is verified.

19.C PROGRAM TO FIND IF A PERSON IS ELIGIBLE FOR VOTING OR NOT

Aim: To write a c program to find if a person is eligible for voting or not

Algorithm:

1: Start
2: Accept the age of the person.
3: If age is greater than or equal to 18, then display 'You are eligible to vote'.
4: If age is less than 18, then display 'You are not eligible to vote'.
5: Stop

Program:

```
#include <stdio.h>
int main()

{
 int age;
printf("Enter age : ");
 scanf("%d", &age);
 if (age >= 18)
 printf("You can Vote!");
 else
 printf("You cant Vote!");
  return 0;
}
```

Output:

Enter your age: 18
You can vote

Enter your age:16
You cant vote.

Result: The program is compiled, executed and the output is verified.

20. C PROGRAM TO FIND WHETHER A NUMBER IS ARMSTRONG OR NOT

Aim: C program to find whether a number is armstrong or not

Algorithm:

Declare variables sum, temp, n, r.
Take the value of n from the user.
Initialize variable sum to 0 and take backup of n as temp = n.
Repeat steps 5 - 7 while n > 0.
r = n % 10.
sum = sum + cube of each digit (r × r × r)
n = n / 10.

Program:

```
#include<stdio.h>
int main(){
  int number,remainder,total=0,temp;
  printf("enter the number=");
  scanf("%d",&number);
  temp=number;
  while(number>0){
    remainder=number%10;
    total=total+(remainder*remainder*remainder);
    number=number/10;
  }
  if(temp==total)
    printf("This number is Armstrong number");
  else
    printf("This number is not Armstrong number");
  return 0;
}
```

Output:

```
enter the number=371
This number is Armstrong number
```

enter the number=53

This number is not Armstrong number

Result: The program is compiled, executed and the output is verified.