Algorithm :-

Step 1 : Start

Step 2 : declare Required variables and initialize sum = 0, rev = 0

Step 3 : get "the values for variable "num".

Step 4 : using while loop, perform Step 5 until condition become false.

Step 5 : calculate : d = num % 10,
Num = num /10
Sum = sum + d
Rev = rev * 10 + d

Step 6 : print the values of sum and reverse.

Step 7 : Stop.

Output :-

Enter the number : 52

Sum of digit = 7

Reverse of the number = 25

Aim :- program to find the sum of digits and reverse of a number.

SOURCE CODE :-

```
# include <stdio.h>
# include <conio.h>
void main ()
{
    clrscr();
    int num, sum = 0, rev = 0, d;
    printf("Enter the number:");
    scanf("%d", &num);
    while (num) {
        d = num % 10;
        sum = sum + d;
        num = num /10;
        rev = rev * 10 + d;
    }
    printf("Sum of digit = %d", sum);
    printf("\n Reverse of the number = %d", rev);
    getch();
}
```

Algorithm :-

Step 1 : Start

Step 2 : Declare required variables

Step 3 : Enter the limit n.

Step 4 : Using for loop perform
         Step 5 to until condition
         become false.

Step 5 : Find fibonacci numbers by
         next = first + second ;
         first = second ;
         second = next ;

Step 6 : Print n fibonacci numbers

Step 7 : Stop.

Page No.

Prog. No.

Date.

AIM ➤ Find first n fibonacci numbers.

SOURCE CODE :-

```c
# include <stdio.h>
# include <conio.h>
void main ()
{
    int n, first = 0, second = 1, next , c ;
    clrscr();
    printf (" Enter the numbers of terms:");
    scanf (" %d ", &n);
    printf ("First %d terms of fibonacci
            series are :- ", n);
    for (c = 0 ; c < n ; c++)
    {
        if (c <= 1)
            next = c ;
        else
        {
            next = first + second ;
            first = second ;
            second = next ;
        }
        printf (" %d \n ", next);
    }
}
```

```
}
    getch();
}
```

Output :-

Enter the number of terms : 5

First 5 terms of fibonacci series are:

0
1
1
2
3

Algorithm :-

Step 1 : Start

Step 2 : declare required variables and initialize k=0.

Step 3 : get numbers of rows

Step 4 : using for loop "i" with intial value 0 and condition i<=rows perform

Step 5 : using another for loop "space" with intial value 1 and space <=rows-i

Step 6 : perform step 6.

Step 6 : Perform Print function, Print (" ")

Step 7 : using while loop with k|=2*i-1, Print "*" and increment k values.

Step 8 : Print next line of pyramid by printf ("\n")

Step 9 : Stop.

Output :-

Enter number of rows : 5

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

AIM :-

Program to create a pyramid using '*'.

SOURCE CODE :-

```c
#include<stdio.h>
int main()
{
    int i, j, space, rows, k=0;
    printf("Enter number of rows");
    scanf("%d", &rows);
    for(i=1; i<=rows; i+i; k=0)
    {
        for(space=1; space<=rows-i; i+space)
        {
            printf(" ");
        }
        while(k!=2*i-1)
        {
            printf("*");
            ++k;
        }
        printf("\n");
    }
    return 0;
}
```

Algorithm :-

Step 1 : Start

Step 2 : Read number n

Step 3 : Set f=0

Step 4 : for i=2 to n-1

Step 5 : if n mod i=0 ,then

Step 6 : set f=1 and break.

Step 7 : loop

Step 8 : if f=0 then
        print "The given number is prime"
        else
        print "The given number is
              not prime".

Step 9 : Stop.

AIM :- Check whether a number is
       prime or not.

SOURCE CODE :-

```
#include <stdio.h>
#include <conio.h>
void main ()
{
    clrscr();
    int n, i, f=0;
    printf ("Enter the number :");
    scanf ("%d", &n);
    for (i=2; i<n ; i++)
    {
        if (n %o i ==0)
        {
            f=1;
            break ;
        }
    }
    if (f==0)
        printf ("The given number is
                 prime");
    else
```

```
          printf ("The given number is not prime");
      }
      getch ();
}
```

Output :-

Enter the number : 2
The given number is prime.
Enter the number : 4
The given number is not prime.

Algorithm :-

Step 1 : Start

Step 2 : declare functions like matrix input(), matrixprint(), matrix transpose() with attributes and variables.

Step 3 : declare 2D matrix mat1[][] and values to mat1[][]

Step 4 : get values to mat1[][] using matrixInput().

Step 5 : perform transpose of matrix using matrix transpose with arguments mat1 and print the transpose of matrix mat1.

Step 6 : Stop.

AIM :- Perform matrix transpose

SOURCE CODE :-

```
# include <stdio.h>
# include <conio.h>
# define Row 2
# define col 2

void matrixInput (int mat[][col]);
void matrix print (int mat[][col]);
void matrix Transpose (int mat[][col]);

void main ()
{
    int mat1 [Row][col];
    int mat2 [Row][col];
    int product [Row][col], add [Row][col],
                            sub [Row][col];
    clrscr ();
    printf("Enter elements in metrix of
            size %dx%d \n", Row, col);
    matrixInput (mat1);
    printf("\n matrix before transpose \n");
    matrix print (mat1);
    matrix Transpose (mat1);
    getch ();
```

output :-

Enter elements in metrix of size 2x2

1
2
3
4

matrix before transpose :

1 2
3 4

matrix after transpose :

1 3
2 4.

```
}
void matrix Input (int mat[] [col])
{
    int row, col;
    for (row = 0; row < ROW; row++)
    {
        for (col = 0; col < col; col++)
        {
            scanf ("%d", (*(mat + row) + col));
        }
    }
}

void matrix print (int, mat[] [col])
{
    int row, col;
    for (row = 0; row < ROW; row++)
    {
        for (col = 0; col < col; col++)
        {
            printf ("%d", *(*(mat + row) + col));
        }
        printf ("\n");
    }
}

void matrix Transpose (int mat[] [col])
{
```

```
int  row      ,col,trans[Row][col];
for (row = o ; Row < Row ; row++)
{
for(col = o ; col < col ; col++)
{
*(*(trans+col)+row) = *(*(mat+row)+col);
}
}
printf("matrix after transpose \n");
for(row=o; row<Row; row++)
{
for(col=o; col<col; col++)
{
printf("%d", *(*(trans+row)+col));
}
printf("\n");
}
```

Algorithm :-

Step 1 : Start

Step 2 : The user-defined function
will divide the
denomination into 2000, 500, 100, 50,
amount in to 2000, 500, 100, 50,
20, 10, 5, 2, 1 rupees notes.

Step 3 : treat the amount.

Step 4 : using for loop with limit
size find numbers of
notes and denomination
using formulas.

Step 5 : print denomination of
given numbers.

Step 6 : Stop

AIM : Find the currency denomination
of a given amount.

SOURCE CODE :-

```c
#include<stdio.h>
#include<conio.h>
#define SIZE 9
void main()
{
    int amount, notes, i;
    int denomination[SIZE]={2000,500,
    200,100,50,20,10,5,1};
    clrscr();
    printf("Enter amount:");
    scanf("%d", &amount);
    printf("\n");
    for(i=0; i<SIZE; i++)
    {
        notes = amount/denomination[i]
        if(notes)
        {
            amount = amount % denomination[i];
            printf("%d * %d = %d\n",
            notes, denomination[i],
```

Output :-

Enter amount : 254321

-3 * 2000 = -6000
-3 * 500 = -1500
-1 * 200 = -200
-1 * 100 = -100
-1 * 20 = -20
-3 * 1 = -3

```
            notes * denomination [i]);
        }
    getch();
}
```

Algorithm :-

Step 1 : Start

Step 2 : declare and get a value for variable "year".

Step 3 : if year%4==0 , year%100==0)
and year%400==0) is true
then print leap year,
else print not a leap year.

Step 4 : Stop.

Aim : check for leap year.

SOURCE CODE :-

```
#include <stdio.h>
#include <conio.h>
void main ()
{
    int year;
    printf("Enter an year :");
    scanf("%d", &year);
    if (year%4==0)
    {
        if (year%100==0)
        {
            if (year%400==0)
            {
                printf("%d is a leap year", year);
            }
            else
                printf("%d is not a leap year", year);
        }
        else
            printf("%d is a leap year", year);
        else
```

```
    printf (" %d is not a leap year ", year);
}
```

Output :-

Enter an year : 2020
2020 is an leap year.

Algorithm :-

Step 1 : Start

Step 2 : declare a char array S[200] and other required variables.

Step 3 : get the string value "s".

Step 4 : using for loop with initial value zero and condition S[i] != '\0',

Step 5 : Check condition S[i] == " ",
Perform step 5.
if true perform count ++.

Step 6 : print total words in given string

Step 7 : Stop.

Output :-

Enter the string :
welcome to vims
Number of words in given string are : 3

AIM : Find the number of words in a Sentence.

Source Code :-

```
#include <stdio.h>
#include <string.h>
void main()
{
    char S[200];
    int count=0, i;
    printf("Enter the string : \n");
    scanf("%[^\n]s ",s);
    for(i=0; s[i] != '\0'; i++)
    {
        if(S[i]==' ')
        {
            count ++;
        }
        printf("Number of words in given string are : %d \n", count +1);
    }
}
```

Algorithm :-

Step 1) Start.

Step 2: get the values for 'n'

Step 3: call the function "series()" with
pass by value 'n' and assign the
return value to 'res'.

Step 4: within the function series(),
declare the required variables and
assign sums = 0.0.

Step 5: perform ser = 1/pow(i,i) using
for loop 'i'.

Step 6: calculate sums += ser.

Step 7: return 'sums' value to main()

Step 8: stop.

AIM : Find the sum of the series =

$$1 + \left(\frac{1}{2}\right)2 + \left(\frac{1}{3}\right)3 + \ldots \ldots \text{ to } 0.0001\% \text{ accuracy}$$

SOURCE CODE :-

```
#include<math.h>
#include<stdio.h>
double series (int n)
{
    int i;
    double sums=0.0, ser;
    for(i=1; i<n; i++)  {
    ser = 1/pow(i,i);
    sums += ser;
    }
    return sums;
}
int main()
{
    int n;
    printf("Enter the limit : ");
    scanf("%d", &n);
    double res = series(n);
    printf("\n Sum of the series is %f",
    res);
```

Output :-

Enter the limit :- 5

Sum of the series is 1.291263

return 0;

}

Algorithm :-

step 1 : Start

step 2 : Get String example "Computer Science idea".

step 3 : Find length of the String using strlen () function.

step 4 : using for loop with limit String length find first letters of the words in given String.

step 5 : print abbreviation of String

step 6 : stop.

Output :-

Enter any String :
Computer Science Ideal

c s I.

Aim :- Display the Short form of a String. Eg. Computer Science :- cs.

SoURCE CoDE :-

```
#include<stdio.h>
#include<string.h>
void main ()
{
  char str[100], * ptr, i, l;
  clrscr();
  printf(" Enter any String :\n ");
  gets (str);
  l = strlen(str);
  ptr = str;
  printf(" %c ", *(ptr +0));
  for(i=1; i<l; i++)
  {
    if (*(ptr +i-1) == ' ')
    printf(" %c ", *(ptr +i));
  }
  getch();
}
```

Algorithm :-

Step 1 : Start

Step 2 : get values for variable N.

Step 3 : Using for loop as special case, to calculate $i*=10$ by implementing special loop structure.

Step 4 : Using another for loop with initial values $i = i/10$ and condition $n > 10$

Perform step 5 repeatedly by increment Statement $i / = 10$.

Step 5 : Print the values of n and also perform $n \% = i$.

Step 6 : Stop.

AIM : Create a pattern with the numbers N.

SOURCE CODE :-

```
#include<stdio.h>
#include<conio.h>
void main ()
{
    long n, i=1;
    clrscr();
    printf("Enter number:");
    scanf("%ld", &n);
    printf("\n pattern:\n");
    for(i=10; i<n; i*=10)
    for(i=i/10; n>o; i/=10)
    {
        printf("%ld%ld\n",n);
        n%=i;
    }
    getch();
}
```

Output :-

Enter number : 39174

pattern : 39174

9 1 7 4
1 7 9
7. 9
9

Algorithm :-

Step 1 :- Start

Step 2 : get values for variables 'low'
and 'high'.

Step 3 : using for loop with initial
values 'low' and limit 'high'
perform Step 4 to 8.

Step 4 : assign values of 'i' to temp2
and temp 1.

Step 5 : using while loop with condition
temp1 != 0 perform. temp l=10
and increment 'n' values.

Step 6 : using while loop with condition
temp2 != 0 , perform remainder =
temp % 10.

Result = pow (remainder, n)
temp2 /= 10

Step 7 : If result values' equal to 'i',
Print values of 'i'.

Step 8 : assign and result values
with zero.

Step 9 : Stop.

AIM : Find the Armstrong number.

SOURCE CODE :

```c
#include <stdio.h>
#include <math.h>
int main ()
{
    int low, high, i, temp1, temp2,
        remainder, n=0, result=0;
    printf ("Enter 2 numbers :");
    scanf ("%d %d", &low, &high);
    printf ("Armstrong numbers between
             %d and %d are:", low, high);
    for (i=low+1; i<high; ++i)
    {
        temp2 = i;
        temp1 = i;
        while (temp1 != 0)
        {
            temp1 /=10;
            ++n;
        }
        while (temp2 != 0)
        {
```

Output :-

Enter 2 numbers :

10

500

Armstrong number between 10 and 500 are

153

370

371

407

```
        remainder = temp2 % 10;
        result += pow(remainder, n);
        temp2 /= 10;
    }
    if (result == i) {
        print(" %d ", i);
    }
    n = 0;
    result = 0;
}
return 0;
}
```