Algorithm :-

```
Step1 : Start
Step2 : declare required variables
        and initialize sum=0, rev=0
Step3 : get the values for variable
        "num".
Step4 : using while loop, perform
        step5 until condition become
        false.
Step5 : calculate : d= num % 10,
        Num = num /10
        Sum = Sum+d
        Rev= rev * 10 +d
Step6 : print the values of sum
        and reverse.
Step 7 : Stop.
```

Output :-

```
Enter the number : 52

Sum of digit =7

Reverse of the number = 25
```

Prog. No.........................

Date.........................

Page No.........................

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 digits = %d" ,sum);
 printf ("\n Reverse of the number=
                          %d ", rev);

 getch() ;

}
```

Prog. No............

Date............

Page No............

Algorithm :-

```
Step 1 : Start
Step 2 : Declare required variables
Step 3 : Get 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.
```

AIM :- Find first n fibonacci numbers.

SOURCE CODE :-

```
#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", n );
 for (c=0 ; c<n ; c++)
 {
    if (c<=1)
       next = c ;
    else
    {
       next = first + second;
       first = second ;
       second = next;
    }
    printf("%d /n", next);
```

Output :-

```
Enter the number of terms : 5
First 5 terms of fibonacci series are:
0
1
1
2
3
```

Prog. No..............................

Date.................................. Page No..............................

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

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 to 8.
Step 5 : using another for loop "space" with
         intial value 1 and space <=rows-i
         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

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

Prog. No.................

Date.................   Page No.................

AIM :- Program to create a pyramid using '*'.

SOURCE CODE :-

```
#include<stdio.h>
int main()
{
 int, i, space, rows, k=0;
 printf("Enter number of rows:");
 scanf("%d", &rows);
 for(i=1; i<=rows; ++i; k=0)
 {
  for(space=1; space<=rows-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 1=0 ,then
Step 6 : set f=1 and break.
Step 7 : Loop
Step 8 : if f=0 they
         print~~f~~ "The given number is prime"
         else
         print "The given number is
                             not prime".
Step 9 : Stop.
```

Prog. No.............................

Date............................. Page No.............................

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 % i==0)
    {
      f=1 ;
      break;
    }
 }
 if (f==0)
    Printf ("The given number is
                  prime");
 else
```

Output :-

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

Prog. No.............................

Date.............................

Page No.............................

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

Algorithm :-

```
Step 1 : Start
Step 2 : declare functions like matrix
         input (), matrixprint (), matrix
         transpose() with attributes
         and variables.
Step 3 : declare 2D matrix mat1[][]
Step 4 : get values to mat1[][]
         using matrixInput().
Step 5 : perform transpose of matrix
         using matrixtranspose with
         arguements mat1 and print
         the transpose of matrix mat1.
Step 6 : Stop.
```

AIM : perform metrix transpose

SOURCE CODE :-

```
#include<stdio.h>
#include<conio.h>
#define ROW 2
#define COL 2
void matrixInput(int mat[][COL]);
void matrixprint(int mat[][COL]);
void matrixTranspose(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");
 matrixprint(mat1);
 matrixTranspose(mat1);
 getch();
```

Output :-

```
Enter elements in metrix of size 2 x2
 1
 2
 3
 4
metrix before transpose :
 1  2
 3  4
matrix after transpose :
 1  3
 2  4.
```

Prog. No.............

Date............. Page No.............

```
}
void matrixInput (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])
{
```

Prog. No.............................

Date.............................   Page No.............................

```
int row ,col,trans[Row][COL];
for (row=0; row< Row ; row++)
{
for (col=0 ; col<COL ; col++)
{
*(*(trans+col)+row)=*(*(mat+row)+col));
}
}
printf("matrix after transpose \n");
for(row=0; row<ROW ; row++)
{
 for (col=0 ; col<COL ;col++)
 {
 printf ("%d",*(*(trans+row)+col));
 }
  printf ("\n");
  }
   }
```

Algorithm :-

```
Step 1 : Start
Step 2 : The user-defined function
         will divide the
         denomination
         amount into 2000, 500, 100, 50,
         20, 10, 5, 2, 1 rupees notes.
Step 3 : Get 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 :-

```
#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
```

Prog. No.................................

Date.................................

Page No.................................

```
            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.
```

Prog. No.

Date

Page No.

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
```

Output :-

```
Enter an year :2020
2020 is an leap year.
```

Prog. No..............................

Date..................................

Page No..................................

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

Algorithm :-

```
step 1 : start
step 2 : declare a char array s[200]
         and other required variables.
         and the string value "s".
step 3 : get the string value "s".
step4 : using for loop with initial value
        zero and condition s[i] != '\0',
        perform step5.
step 5 : check condition s[i] == " ",
         if true perform count++.
step 6 : print total words in given
         string
step 7 : stop.
```

Output :-

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

Prog. No.................

Date.................

Page No.................

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 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 = $1 + (\frac{1}{2})2 + (\frac{1}{3})3 + \ldots\ldots$ to 0.0001% 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
```

Prog. No.................................

Date...

```
    return 0;
}
```

Algorithm :-

```
step 1 : ~~step~~ start
step 2 : Get String example "Computer
         science ideal".
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.
```

Prog. No.........................

Date.........................

Page No.........................

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
         intial 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.
```

Output :-

```
Enter number : 39174
pattern : 39174.
   9  1  7  4
 1  7  9
      7  9
        9
```

Prog. No..................

Date..................

Page No..................

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>0 ; i/=10)
 {
 printf("%ld\n", n);
 n %= i;
 }
 getch();

 }
```

Algorithm :-

```
Step 1 : Start
Step 2 : get values for variables 'low'
         and 'high'.
Step 3 : using for loop with intial
         values 'low' and limit 'high'
         perform Step 4 to 8.
Step 4 : assign values of 'i' to temp2
         and temp1.
Step 5 : using while loop with condition
         temp1 != 0 perform temp /= 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 1,
         print values of 'i'
Step 8 : assign and result values
         with zero.
Step 9 : Stop.
```

AIM : Find the Armstrong number.

SOURCE CODE :

```
#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 ;

}
```

Algorithm :-

```
Step 1 : Start
Step 2 : declare file pointer 'all' 'even' and 'odd'
         also declare required variables.
Step 3: get total number of records to
        variable 'number'.
Step 4: using fopen(), open file 'ANYNUMBER' in
        write mode and assign value to
        file pointer 'all'.
Step 5: using for loop with limit records,
        get value to variable 'number'.
Step 6: if value of number = = -1 , break.
Step 7: write number to file pointer 'all'
        using putw().
Step 8: close all files using fclose()
Step 9: open 'ANYNUMBER' 'ODDNUMBER 'EVEN
        NUMBER files in read and write
        mode .
Step 10 : calculate odd and even number
          and assign values to filepointes
          'even' and 'odd'.
Step 11 : Close all files.
Step 12 : print even and odd numbers
          from respective files.
Step 13 : Stop.
```

Prog. No.................................

Date.................................

Page No.................................

AIM : write odd and even numbers into Separate files.

SOURCE CODE :-

```
#include<stdio.h>
#include<math.h>
void main() {

FILE *all, *even, *odd;
int number , i , records;
clrscr();
printf("INPUT THE TOTAL NUMBER OF
        RECORDS THAT U WANT TO
        ENTER:");
scanf("%d", &records);
printf(" even numbers ");
all = fopen("ANY NUMBER","w");
for (i=1 ; i<=records ; i++)
{
scanf(" %d ", &number);
if(number ==-1) break;
putw(number, all);

}
```

Output :-

```
INPUT THE TOTAL NUMBER OF RECORDS THAT
U WANT To ENTER :
5
Enter number :
1
2
3
4
5
THE EVEN NUMBERS ARE :
2
4
THE ODD NUMBERS ARE :
1
3
5
```

Prog. No.............................

Date.............................

Page No.............................

```
fclose (all);
all = fopen ("ANY NUMBER ","r");
even=fopen ("EVEN NUMBER","w");
odd =fopen ("ODD NUMBER", "w");
while (number=getw(all) != EOF)
{
if (number %2==0)
putw(number, even);
else
putw (number, odd);
}
fclose(all);
fclose (even);
fclose (odd);
even =fopen ("EVEN NUMBER", "r");
odd =fopen ("ODD NUMBER","r");
printf ("THE EVEN NUMBERS Are :");
while((number = getw (even) != EOF)
printf ("\n %d", number);
printf ("\n THE ODD NUMBERS ARE:");
while((number = getw (odd) != EOF )
printf (" \n %d \n", number);
fclose(even);
fclose (odd);
 getch ();

}
```

Algorithm :-

```
Step 1 :- Start
Step 2 :- declare required variables and
          get the values for 'String 1'.
Step 3 :- calculate length of String using
          Strlen () and assign the value
          to variable 'length'.
Step 4 :- using for loop with limit 'length'
          perform Step 5.
Step 5 :- Set flag = 1, if the condition
          String [i], String 1 [length -i-1] is
          true.
Step 6 :- if flag value is 0, print not
          palindrome, else print
          palindrome.
Step 7 :- Stop.
```

AIM :- Check for palindrom String.

SOURCE CODE :-

```
#include <Stdio.h>
#include <String.h>
int main () {
 Char String 1 [20];
 int i, length;
 int flag = 0;
 printf ("Enter a String :");
 Scanf ("%s", String 1);
 length = Strlen( String 1);
 for (i=0; i<length; i++)
 {
  if (String 1 [i] != String 1 [length -i-1])
  {
   flag = 1;
   break;
  }
 }
  if (flag) {
     printf (" %s is not palindrom",
                         String 1);
  }
```

Output :-

```
Enter a string :
malayalam .
malayalam is a palindreme.
```

Prog. No............................

Date..................................

Page No..............................

```
    else {
        printf("%s is a palindreme",
                         string1);
    }
    return 0;
}
```