

**Week 1:**

1. Write a C program to find the sum of individual digits of a positive integer.
2. Fibonacci sequence is defined as follows: the first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Write a C program to generate the first n terms of the sequence.
3. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.
4. Write a C program to find the roots of a quadratic equation.

**Week 2:**

5. Write a C program to find the factorial of a given integer.
6. Write a C program to find the GCD (greatest common divisor) of two given integers.
7. Write a C program to solve Towers of Hanoi problem.
8. Write a C program, which takes two integer operands and one operator from the user, performs the operation and then prints the result. (Consider the operators +, -, \*, /, % and use Switch Statement)

**Week 3:**

9. Write a C program to find both the largest and smallest number in a list of integers.
10. Write a C program that uses functions to perform the following:
  - i) Addition of Two Matrices
  - ii) Multiplication of Two Matrices

**Week 4:**

11. Write a C program that uses functions to perform the following operations:
  - i) To insert a sub-string in to a given main string from a given position.
  - ii) To delete n Characters from a given position in a given string.
12. Write a C program to determine if the given string is a palindrome or not
13. Write a C program that displays the position or index in the string S where the string T begins, or - 1 if S doesn't contain T.
14. Write a C program to count the lines, words and characters in a given text.

**Week 5:**

15. Write a C program to generate Pascal's triangle.
16. Write a C program to construct a pyramid of numbers.
17. Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression:  $1+x+x^2+x^3+\dots+x^n$   
For example: if n is 3 and x is 5, then the program computes  $1+5+25+125$ .  
Print x, n, the sum  
Perform error checking. For example, the formula does not make sense for negative exponents – if n is less than 0. Have your program print an error message if  $n < 0$ , then go back and read in the next pair of numbers of without computing the sum. Are any values of x also illegal? If so, test for them too.

**Week 6:**

18. 2's complement of a number is obtained by scanning it from right to left and complementing all the bits after the first appearance of a 1. Thus 2's complement of 11100 is 00100. Write a C program to find the 2's complement of a binary number.

19. Write a C program to convert a Roman numeral to its decimal equivalent.

**Week 7:**

20. Write a C program that uses functions to perform the following operations:
  - i) Reading a complex number
  - ii) Writing a complex number
  - iii) Addition of two complex numbers
  - iv) Multiplication of two complex numbers(Note: represent complex number using a structure.)

**Week 8:**

21. i) Write a C program which copies one file to another.  
ii) Write a C program to reverse the first n characters in a file. (Note: The file name and n are specified on the command line.)
22. i) Write a C program to display the contents of a file.  
ii) Write a C program to merge two files into a third file (i.e., the contents of the first file followed by those of the second are put in the third file)

**Week 9:**

- i) Write a C program that implements the following sorting methods to sort a given list of integers in ascending order i) Bubble sort ii) Selection sort iii) Insertion sort

**Week 10:**

27. Write C programs that use both recursive and non recursive functions to perform the following searching operations for a Key value in a given list of integers:
  - i) Linear search ii) Binary search

**1) Program :**

**// Sum of Individual Digits of an Number**

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

```
int main()
```

```
{
```

```
    int num,last,sum = 0,temp;
```

```
    printf("Enter a Number : ");
```

```
    scanf("%d",&num);
```

```
    temp = num;
```

```
    while(num != 0)
```

```
    {
```

```
        last = num%10;
```

```
        sum = sum + last;
```

```
        num = num/10;
```

```
    }
```

```
    printf("Sum of the Digits of %d is %d..",temp,sum);
```

```
    return 0;
```

```
}
```

**Output :**

**Enter a Number : 53**

**Sum of the Digits of 53 is 8..**

**2) Program :**

**// generate a Fibonacci Series upto nth number**

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

```
int main()
```

```
{
```

```
    int num, a = 0, b = 1, c, count;
```

```
    printf("Enter a number to Generate a Fibonaci Series : ");
```

```
    scanf("%d",&num);
```

```
    printf("The fibonacci series : \n");
```

```
    printf("%d %d ",a,b);
```

```

count = 2;
while(count < num)
{
    c = a + b;
    a = b;
    b = c;
    printf("%d ",c);
    count++;
}
return 0;
}

```

**Output :**

**Enter a number to Generate a Fibonacci Series : 8**

**The fibonacci series :**

**0 1 1 2 3 5 8 13**

**3) Program :**

**// Generate all prime numbers from 1 to n**

**#include<stdio.h>**

```

int main()
{

    int num,i,count,n;
    printf("Enter max range to get all prime numbers : ");
    scanf("%d",&n);

    for(num = 1; num <= n; num++){

        count = 0;

        for(i = 2; i <= num/2; i++)
        {
            if(num%i == 0)
            {
                count++;
                break;
            }
        }

        if(count==0 && num!= 1)
        {
            printf("%d ",num);
        }
    }
}

```

```
}  
  
return 0;  
}
```

**Output :**

**Enter max range to get all prime numbers : 50**

**2 3 5 7 11 13 17 19 23 29 31 37 41 43 47**

**4) Program :**

**// Solving the roots of Quadratic equation**

**# include<stdio.h>**

**# include<math.h>**

```
int main()  
{  
    float a,b,c,r1,r2,d;  
    printf("enter the values of a b c : ");  
    scanf("%f %f %f",&a,&b,&c);  
  
    d = (b*b) - (4*a*c);  
    if (d > 0)  
    {  
        r1 = ((-b) + sqrt(d)/(2*a));  
        r2 = ((-b) - sqrt(d)/(2*a));  
        printf("The real roots = %f %f", r1, r2);  
    }  
    else if (d == 0)  
    {  
        r1 = (-b)/(2*a);  
        r2 = (-b)/(2*a);  
        printf("roots are equal =%f %f", r1, r2);  
    }  
    else  
    {  
        printf("Roots are imaginary");  
    }  
  
    return 0;  
}
```

**Out[put] :**

**enter the values of a b c : 1 2 1**

**roots are equal =-1.000000 -1.000000**

## WEEK - 2

### 1) Program :

// To generate a Factorial of a number

```
#include<stdio.h>

int main()
{
    int n,i,fact = 1;

    printf("Enter a number : ");
    scanf("%d",&n);

    for(i = 1; i <= n; i++)
    {
        fact = fact*i;
    }
    printf("Factorial of a Number : %d",fact);
    return 0;
}
```

Output :

Enter a number : 5

Factorial of a Number : 120

### 2) Program :

// GCD of two numbers;

```
#include <stdio.h>
int main()
{
    int n1, n2, i, gcd;

    printf("Enter two integers: ");
    scanf("%d %d", &n1, &n2);

    for(i=1; i <= n1 && i <= n2; ++i)
    {
        // Checks if i is factor of both integers
        if(n1%i==0 && n2%i==0)
        {
            gcd = i;
        }
    }
}
```

```

printf("G.C.D of %d and %d is %d", n1, n2, gcd);

return 0;
}

```

Output :

Enter two integers: 81 153

G.C.D of 81 and 153 is 9

### 3) Program :

//C program for Tower of Hanoi using Recursion

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

```
void towers(int, char, char, char);
```

```
int main()
```

```

{
    int num;

    printf("Enter the number of disks : ");
    scanf("%d", &num);
    printf("The sequence of moves involved in the Tower of Hanoi are :\n");
    towers(num, 'A', 'C', 'B');
    return 0;
}

```

```
void towers(int num, char frompeg, char topeg, char auxpeg)
```

```

{
    if (num == 1)
    {
        printf("\n Move disk 1 from peg %c to peg %c", frompeg, topeg);
        return;
    }
    towers(num - 1, frompeg, auxpeg, topeg);
    printf("\n Move disk %d from peg %c to peg %c", num, frompeg, topeg);
    towers(num - 1, auxpeg, topeg, frompeg);
}

```

Output :

Enter the number of disks : 3 2

The sequence of moves involved in the Tower of Hanoi are :

Move disk 1 from peg A to peg B

Move disk 2 from peg A to peg C

Move disk 1 from peg B to peg C

4) Program :

// Calculator Program

```
#include<stdio.h>
int main()
{
    char choice;
    float num1,num2,result;
    int flag=1;

    printf("Enter +,-,/,* for knowing the result : \n");
    scanf("%c",&choice);
    printf("Enter number 1 : \n");
    scanf("%f",&num1);
    printf("Enter number 2 : \n");
    scanf("%f",&num2);

    switch(choice)
    {
        case '+':
            result = num1+num2;
            break;
        case '-':
            result = num1-num2;
            break;
        case '/':
            {
                if(num2==0)
                {
                    flag=0;
                }
                else
                {
                    result = num1/num2;
                }
            }
            break;
        case '*':
            result = num1*num2;
            break;
        default:
            printf("Error");
            break;
    }
}
```

```

    if(flag==1)
    {
        printf("%f %c %f = %f",num1,choice,num2,result);
    }
    else
    {
        printf("%f %c %f = undefined\n",num1,choice,num2);
    }

    return 0;
}

```

Output :

Enter +,-,/,\* for knowing the result :

-

Enter number 1 :

12

Enter number 2 :

2

12.000000 - 2.000000 = 10.000000

### WEEK - 3

1) Program :

// C program to find the smallest and largest element in an array

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

```
int main()
```

```
{
```

```
    int a[50],i,n,large,small;
```

```
    printf("\nEnter the number of elements : ");
```

```
    scanf("%d",&n);
```

```
    printf("\nInput the array elements : ");
```

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

```
    {
```

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

```
    }
```

```
    large=small=a[0];
```

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

```
    {
```

```
        if(a[i]>large)
```



```

        {
            large=a[i];
        }

        if(a[i]<small)
        {
            small=a[i];
        }
    }

    printf("\nThe smallest element is %d\n",small);
    printf("\nThe largest element is %d\n",large);

    return 0;
}

```

Output :

Enter the number of elements : 5

Input the array elements : 1 5 2 8 9

The smallest element is 1

The largest element is 9

2) Program :

// Function for matrixes

```

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

```

```

void add()
{
    int i,j,rows,col;
    printf("Enter number of rows\n");
    scanf("%d",&rows);
    printf("Enter number of columns\n");
    scanf("%d",&col);

    int a1[rows][col],a2[rows][col],add[rows][col];
    //Taking input for 1st matrix
    printf("Enter Matrix 1 :\n ");
    for(i=0;i<rows;i++)
    {
        for(j=0;j<col;j++)

```

```

        {
            scanf("%d",&a1[i][j]);
        }
    }
    //Taking input for 2nd matrix
    printf("Enter Matrix 2 : \n");
    for(i=0;i<rows;i++)
    {
        for(j=0;j<col;j++)
        {
            scanf("%d",&a2[i][j]);
        }
    }
    //Addition of matrix
    for(i=0;i<rows;i++)
    {
        for(j=0;j<col;j++)
        {
            add[i][j]=a1[i][j]+a2[i][j];
        }
    }

    printf("Addition of above matrices is\n");

    for(i=0;i<rows;i++)
    {
        for(j=0;j<col;j++)
        {
            printf("%d\t",add[i][j]);
        }
        printf("\n");
    }
}

void mul()
{
    int i,j,k,rows_1,col_1,rows_2,col_2,sum=0;
    printf("Enter number of rows and columns of matrix 1\n");
    scanf("%d %d",&rows_1,&col_1);
    printf("Enter number of rows and columns of matrix 2\n");
    scanf("%d %d",&rows_2,&col_2);

    int a1[rows_1][col_1],a2[rows_2][col_2],mul[rows_1][col_2];
    if(col_1==rows_2)
    {
        //Taking input for 1st matrix

```

```

printf("Enter Matrix 1 : ");
for(i=0;i<rows_1;i++)
{
    for(j=0;j<col_1;j++)
    {
        scanf("%d",&a1[i][j]);
    }
}
//Taking input for 2nd matrix
printf("Enter Matrix 2 : ");
for(i=0;i<rows_2;i++)
{
    for(j=0;j<col_2;j++)
    {
        scanf("%d",&a2[i][j]);
    }
}
//multiplication of matrix
for(i=0;i<rows_1;i++)
{
    for(j=0;j<col_2;j++)
    {
        for(k=0;k<rows_2;k++)
        {
            sum+=a1[i][k]*a2[k][j];
        }
        mul[i][j]=sum;
        sum=0;
    }
}

printf("Multiplication of above matrices is\n");

for(i=0;i<rows_1;i++)
{
    for(j=0;j<col_2;j++)
    {
        printf("%d\t",mul[i][j]);
    }
    printf("\n");
}
}
else
{
    printf("Not possible with the given rows and columns\n");
}

```

```

}

int main()
{
    int d;
    printf("Enter 1 ==> For Matrix Addition \n");
    printf("Enter 2 ==> For Matrix Multiplication \n");
    printf("Enter 3 ==> To exit \n");

    printf("Enter your choice : ");
    scanf("%d",&d);
    switch(d)
    {
        case 1:
            add();
            break;
        case 2:
            mul();
            break;
        case 3:
            exit(0);

    }
    return 0;
}

```

Output :

```

Enter 1 ==> For Matrix Addition
Enter 2 ==> For Matrix Multiplication
Enter 3 ==> To exit
Enter your choice : 1
Enter number of rows
2
Enter number of columns
2
Enter Matrix 1 :
1
1 2
3
4
Enter Matrix 2 :
5
6
7
8
Addition of above matrices is

```

6      8  
10     12

#### WEEK - 4

1) Program :

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

void delchar(char *x,int a, int b);

void string()
{
    char string[10];
    int n,pos,p1;

    puts("Enter the string");
    gets(string);
    printf("Enter the position from where to delete");
    scanf("%d",&pos);
    printf("Enter the number of characters to be deleted");
    scanf("%d",&n);
    delchar(string, n,pos);
}

void delchar(char *x1,int a1, int b1)
{
    if ((a1+b1-1) <= strlen(x1))
    {
        strcpy(&x1[b1-1],&x1[a1+b1-1]);
        puts(x1);
    }
}

void add()
{
    char a[10];
    char b[10];
    char c[10];
    int p=0,r=0,i=0;
    int t=0;
    int x,g,s,n,o;

    puts("Enter First String:");
```

```

    gets(a);
    puts("Enter Second String:");
    gets(b);
    printf("Enter the position where the item has to be inserted: ");
    scanf("%d",&p);
    r = strlen(a);
    n = strlen(b);
    i=0;

    // Copying the input string into another array
    while(i <= r)
    {
        c[i]=a[i];
        i++;
    }
    s = n+r;
    o = p+n;

    // Adding the sub-string
    for(i=p;i<s;i++)
    {
        x = c[i];
        if(t<n)
        {
            a[i] = b[t];
            t=t+1;
        }
        a[o]=x;
        o=o+1;
    }

    printf("%s", a);
}

void main()
{
    int d;
    printf("Enter 1 ==> For To insert a string into a main string \n");
    printf("Enter 2 ==> For To delete n characters from the string \n");
    printf("Enter 3 ==> To exit \n");

    printf("Enter your choice : ");
    scanf("%d",&d);
    switch(d)
    {
        case 1:

```

```

        string();
        break;
    case 2:
        add();
        break;
    case 3:
        exit(0);
    }
}

```

## 2) Program :

// to check string is palindrome or not

```

#include <stdio.h>
#include <string.h>

int main()
{
    char string1[20];
    int i, length;
    int flag = 0;

    printf("Enter a string:");
    scanf("%s", string1);

    length = strlen(string1);

    for(i=0;i < length ;i++)
    {
        if(string1[i] != string1[length-i-1])
        {
            flag = 1;
            break;
        }
    }

    if(flag) {
        printf("%s is not a palindrome", string1);
    }
    else {
        printf("%s is a palindrome", string1);
    }
    return 0;
}

```

Output :  
Enter a string:malayalam  
malayalam is a palindrome

3) Program :

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

void main()
{
    char s[30], t[20];
    char *found;

    puts("Enter the first string: ");
    gets(s);
    puts("Enter the string to be searched: ");
    gets(t);
    found = strstr(s, t);
    if(found)
    {
        printf("Second String is found in the First String at %d position.\n", found - s + 1);
    }
    else
    {
        printf("-1");
    }
}
```

Output :  
Enter the first string:  
welcome to c  
Enter the string to be searched:  
to  
Second String is found in the First String at 9 position.

4) Program :

// write a c program to count the lines words and characters in a given text

```
#include<stdio.h>
int main()
{
    // declare variables
    char str[200];
    int line, word, ch;
```



```

// initialize count variables with zero
line = word = ch = 0;

// read multiline string
printf("Enter string terminated with ~ :\n");
scanf("%[^~]", str);

// check every character
for(int i=0; str[i]!='\0'; i++)
{
    // if it is new line then
    // one line and one word completed
    if(str[i]=='\n')
    {
        line++;
        word++;
    }

    // else it is a character
    else
    {
        // if character is space or tab
        // then one word is also completed
        if(str[i]==' '||str[i]=='\t')
        {
            word++;
            ch++;
        }

        // it was not '\n', sapace or tab
        // it is a normal character
        else {
            ch++;
        }
    }
}

// display count values
printf("\nCharacter counts = %d\n", ch);
printf("Word counts = %d\n", word);
printf("Line counts = %d\n", line);

return 0;
}

```

Output :  
Enter string terminated with ~ :  
welcome to c-language ~

Character counts = 22  
Word counts = 3  
Line counts = 0

## WEEK - 5

### 1) Program :

// c program to generate a Pascal Triangle

```
#include<stdio.h>
long factorial(int);

int main()
{
    int i, n, c;
    printf("Enter the number of rows you wish to see in pascal triangle : ");
    scanf("%d", & n);

    for(i = 0; i < n; i++)
    {
        for(c = 0; c <= (n - i - 2); c++){ printf(" "); }

        for(c = 0; c <= i; c++){ printf("%ld ", factorial(i) / (factorial(c) * factorial(i - c))); }
        printf("\n");
    }
    return 0;
}

long factorial(int n)
{
    int c;
    long result = 1;
    for(c = 1; c <= n; c++){ result = result * c; }
    return result;
}
```

Output :  
Enter the number of rows you wish to see in pascal triangle : 5

```
1
1 1
1 2 1
```

```
1 3 3 1
1 4 6 4 1
```

2) Program :

// C program to generate pyramaid of numbers

```
#include <stdio.h>
int main()
{
    int rows, space, i, j;

    printf("Enter number of rows: "); // enter a number for generating the pyramid
    scanf("%d",&rows);

    for(i=0; i<rows; i++) // outer loop for displaying rows
    {
        for(space=1; space <= rows-i; space++) // space for each and every element
        {
            printf(" ");
        }

        for(j=0-i; j <= i; j++) //inner loop for displaying the pyramid of numbers
        {

            printf("%2d",abs(j)); // prints the value

        }
        printf("\n"); // every line in different row
    }
}
```

Output :

Enter number of rows: 5

```
0
1 0 1
2 1 0 1 2
3 2 1 0 1 2 3
4 3 2 1 0 1 2 3 4
```

3) Program :

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

```
int main()
{
```

```

int x,n,sum = 1;

printf("Enter x - value : ");
scanf("%d",&x);

printf("Enter n - value : ");
scanf("%d",&n);

for(int i = 1; i <= n; i++)
{
    sum = sum + pow(x,i);
}
printf("Result : %d",sum);

return 0;
}

```

Output :

Enter x - value : 2

Enter n - value : 2

Result : 7

## WEEK - 6

### 1) Program :

// C program to write 2's complement of the given Binary number

```

#include <stdio.h>
int main()
{
    int n; // variable declaration
    printf("Enter the number of bits do you want to enter :");
    scanf("%d",&n);
    char binary[n+1]; // binary array declaration;
    char onescomplement[n+1]; // onescomplement array declaration
    char twoscomplement[n+1]; // twoscomplement array declaration
    int carry=1; // variable initialization

    printf("\nEnter the binary number : ");
    scanf("%s", binary);
    printf("%s", binary);
    printf("\nThe ones complement of the binary number is :");

    // Finding onescomplement in C

```

```

for(int i=0;i<n;i++)
{
    if(binary[i]=='0')
        onescomplement[i]='1';
    else if(binary[i]=='1')
        onescomplement[i]='0';
}
onescomplement[n]='\0';
printf("%s",onescomplement);

printf("\nThe twos complement of a binary number is : ");

// Finding twoscomplement in C
for(int i=n-1; i>=0; i--)
{
    if(onescomplement[i] == '1' && carry == 1)
    {
        twoscomplement[i] = '0';
    }
    else if(onescomplement[i] == '0' && carry == 1)
    {
        twoscomplement[i] = '1';
        carry = 0;
    }
    else
    {
        twoscomplement[i] = onescomplement[i];
    }
}

twoscomplement[n]='\0';
printf("%s",twoscomplement);
return 0;
}

```

Output :

Enter the number of bits do you want to enter : 5

Enter the binary number : 10100

10100

The ones complement of the binary number is :01011

The twos complement of a binary number is : 01100

2) Program :

// Program to convert Roman Numerals to Numbers

```

#include <stdio.h>
#include <string.h>

// This function returns value of a Roman symbol
int value(char r)
{
    if (r == 'I'){ return 1; }
    if (r == 'V'){ return 5; }
    if (r == 'X'){ return 10; }
    if (r == 'L'){ return 50; }
    if (r == 'C'){ return 100; }
    if (r == 'D'){ return 500; }
    if (r == 'M'){ return 1000; }

    return -1;
}

// Returns decimal value of roman numeral
int romanToDecimal(char str[])
{
    int res = 0;

    for (int i = 0; i < strlen(str); i++)
    {
        int s1 = value(str[i]);

        if (i + 1 < strlen(str))
        {
            int s2 = value(str[i + 1]);

            if (s1 >= s2)
            {
                res = res + s1;
            }
            else
            {
                res = res + s2 - s1;
                i++;
            }
        }
        else
        {
            res = res + s1;
        }
    }
    return res;
}

```

```

}

int main()
{
    char str[10] = "MCMIV";
    printf("Integer form of Roman Numeral is %d",romanToDecimal(str));

    return 0;
}

```

Output :  
Integer form of Roman Numeral is 1904

## WEEK - 7

- 1) Program :  
// reading and writing a complex numbers using Structures

```

#include<stdio.h>

typedef struct complex
{
    int real;
    int img;
} complex;

int main()
{
    complex n1;

    printf("Enter the Real and Imaginary parts : ");
    scanf("%d %d",&n1.real, &n1.img);

    printf("Complex number : %d + i %d",n1.real,n1.img);
    return 0;

}

```

Output :  
Enter the Real and Imaginary parts : 2 3  
Complex number : 2 + i 3

- 2) Program :  
// Add complex numbers using Structures

```

#include<stdio.h>

typedef struct complex
{
    int real;
    int img;
} complex;

complex add(complex n1, complex n2);

int main()
{
    complex n1, n2, result;

    printf("For 1st complex number \n");
    printf("Enter the Real and Imaginary parts : ");
    scanf("%d %d",&n1.real, &n1.img);

    printf("For 2nd complex number \n");
    printf("Enter the Real and Imaginary parts : ");
    scanf("%d %d",&n2.real, &n2.img);

    result = add(n1,n2);

    printf("Sum = %d + i %d",result);
    return 0;
}

complex add(complex n1, complex n2)
{
    complex temp;
    temp.real = n1.real + n2.real;
    temp.img = n1.img + n2.img;
    return (temp);
}

```

Output :

```

For 1st complex number
Enter the Real and Imaginary parts : 2 3
For 2nd complex number
Enter the Real and Imaginary parts : 4 5
Sum = 6 + i 3

```



- 3) Program :  
// Multiply complex numbers using Structures

```
#include<stdio.h>

typedef struct complex
{
    int real;
    int img;
} complex;

complex mul(complex n1, complex n2);

int main()
{
    complex n1, n2, result;

    printf("For 1st complex number \n");
    printf("Enter the Real and Imaginary parts : ");
    scanf("%d %d",&n1.real, &n1.img);

    printf("For 2nd complex number \n");
    printf("Enter the Real and Imaginary parts : ");
    scanf("%d %d",&n2.real, &n2.img);

    result = mul(n1,n2);

    printf("Sum = %d + i %d",result.real,result.img);
    return 0;
}

complex mul(complex n1, complex n2)
{
    complex temp;
    temp.real = n1.real * n2.real;
    temp.img = n1.img * n2.img;
    return (temp);
}
```

Output :

```
For 1st complex number
Enter the Real and Imaginary parts : 2 3
For 2nd complex number
Enter the Real and Imaginary parts : 4 5
```

Sum = 8 + i 15

## WEEK - 8

1) Program :

// file copying

```
#include<stdio.h>
#include<stdlib.h>
Int main ()
{
    FILE *file1,*file2;
    char c;
    int tabs=0,lines=0,spaces=0,characters=0;
    file1=fopen("file7.txt","r");

    if(file1==NULL)
    {
        printf("File 1 Not Found\n");
        exit(0);
    }

    file2=fopen("file8.txt","w");
    if(file2==NULL)
    {
        printf("File 2 Not Found\n");
        exit(0);
    }
    while(1)
    {
        c=fgetc(file1);
        if(c!=EOF)
        {
            fputc(c,file2);
        }
        else
        {
            break;
        }
    }
    fclose(file1);
    fclose(file2);
}
```

Output :

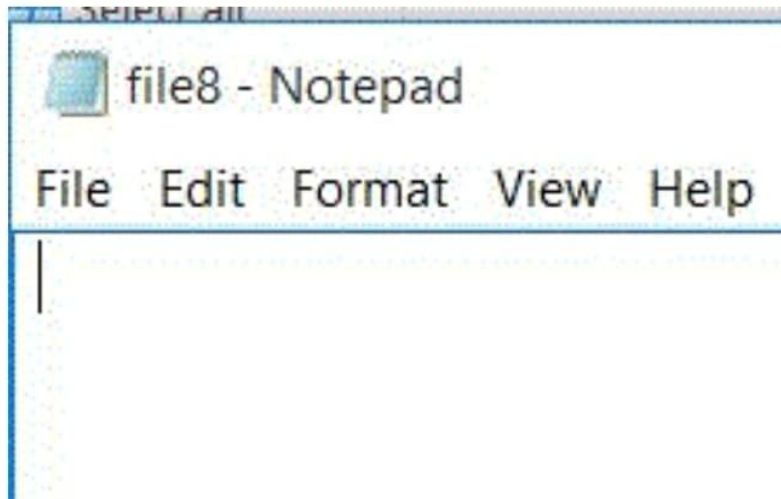
## Before Running Program File 7 and File 8

---

file7 - Notepad

File Edit Format View Help

This is Paragraph for Explaing File Programs  
You can edit this when ever you want  
This is Mahidhar who is the author of cprograms4future blog  
I hope every one of you liked this blog and expect more programs in future  
Thank You



## After Running the Program File 8 has the following text copied from File 7

---

file8 - Notepad

File Edit Format View Help

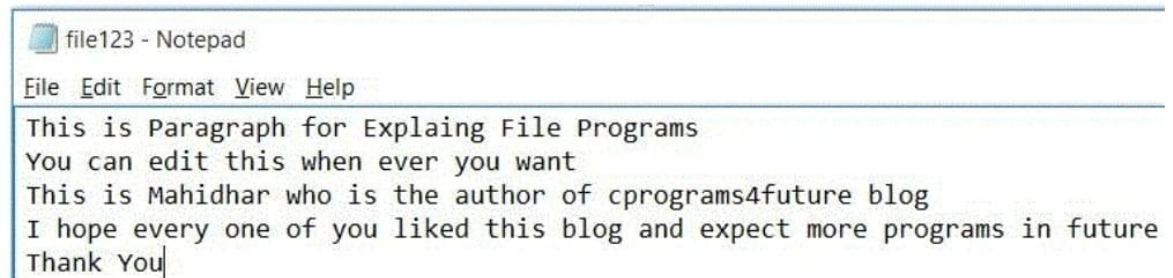
This is Paragraph for Explaing File Programs  
You can edit this when ever you want  
This is Mahidhar who is the author of cprograms4future blog  
I hope every one of you liked this blog and expect more programs in future  
Thank You

```
2) Program :  
#include<stdio.h>  
#include<stdlib.h>  
main ()  
{  
    FILE *file;
```

```
char c;
file=fopen("file123.txt","r");
while(1)
{
    if(file==NULL)
    {
        printf("File Not Found\n");
        exit(0);
    }
    else
    {
        c=fgetc(file);
        if(c==EOF)
        {
            break;
        }
        printf("%c",c);
    }
}
fclose(file);
}
```

Output :

This is the text file(file123) which I saved in one folder:



Text from file123 which is printed on console

```
This is Paragraph for Explaing File Programs
You can edit this when ever you want
This is Mahidhar who is the author of cprograms4fut
I hope every one of you liked this blog and expect
Thank You
```

3) Program :

```
// Merge of files
#include<stdio.h>
#include<stdlib.h>
main ()
{
    FILE *file1,*file2,*mergeFile;
    char c;
    file1=fopen("file1.txt","r");
    file2=fopen("file2.txt","r");

    if(file1==NULL)
    {
        printf("File 1 Not Found\n");
        exit(0);
    }

    if(file2==NULL)
    {
        printf("File 2 Not Found\n");
        exit(0);
    }


    mergeFile=fopen("mergefile.txt","w");
    if(mergeFile==NULL)
    {
        printf("Merge File Not Found\n");
        exit(0);
    }
    while(1)
    {
        c=fgetc(file1);
        if(c!=EOF)
        {
            fputc(c,mergeFile);
        }
        else
        {
            break;
        }
    }
    fputs("\n-----From file 2-----\n",mergeFile);
    while(1)
    {
        c=fgetc(file2);
        if(c!=EOF)
```

```
{  
    fputc(c,mergeFile);  
}  
else  
{  
    break;  
}  
}  
fclose(file1);  
fclose(file2);  
fclose(mergeFile);  
printf("Successfully Merged\n");  
}
```

Output :

## Before Running the Program

---


 file1.txt - Notepad

File Edit Format View Help

---

Hello World From File 1

---

 file2.txt - Notepad

File Edit Format View Help

---

This is CPROGRAMS4FUTURE.COM from file2


 mergefile.txt - Notepad

File Edit Format View Help


---

|

## After Running the Program

 "C:\Users\lenovo\Desktop\wordpress\programs\Files\Prog 390 Merge 2 Files\

```
Successfully Merged  
Press any key to continue . . .
```

 mergefile.txt - Notepad  
File Edit Format View Help  
Hello World From File 1  
-----From file 2-----  
This is CPROGRAMS4FUTURE.COM from file2

## WEEK - 9

1) Program :

### i) **Bubblesort:**

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

```
void bubblesort(int arr[],int n)
```

```
{
```

```
    int temp,i,j;
```



```

        for(int i=0;i<n-1;i++)
        {
            for(int j=0;j<n-i-1;j++)
            {
                if(arr[j] > arr[j+1])
                {
                    temp=arr[j+1];
                    arr[j+1]=arr[j];
                    arr[j]=temp;
                }
            }
        }
    }

int main()
{
    int n,a[10],i;

    printf("Enter the number of terms to sort ");

    scanf("%d",&n);

    for(int i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }

```

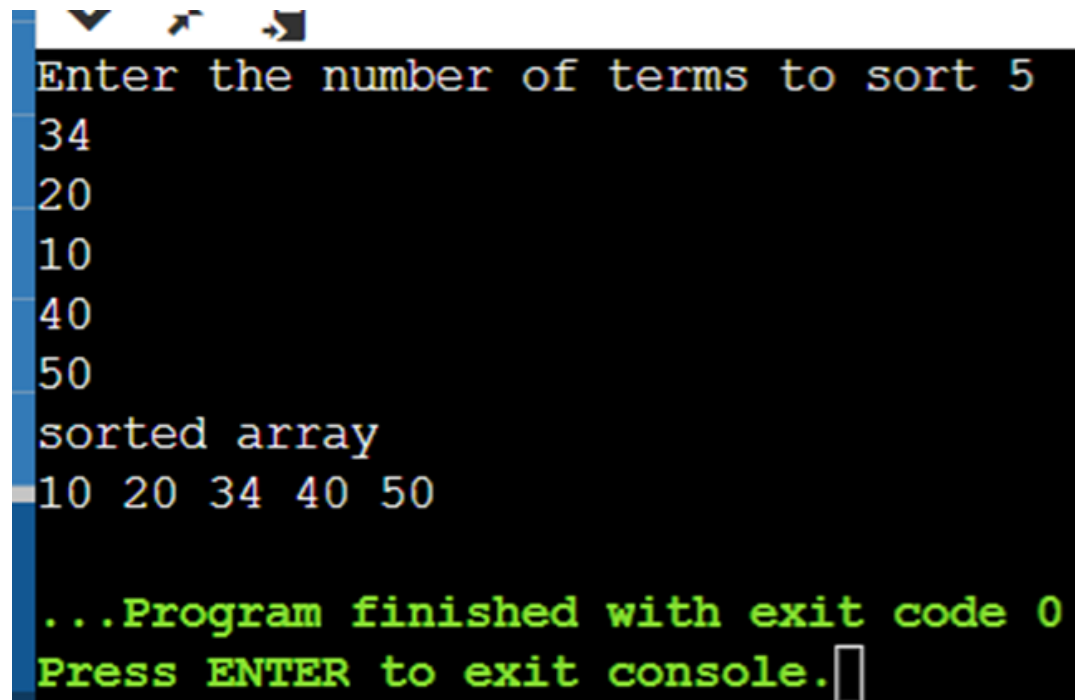
```
bubblesort(a,n);

printf("sorted array\n");

    for(int i=0;i<n;i++)
    {
        printf("%d ",a[i]);
    }

    return 0;
}
```

Output :



```
Enter the number of terms to sort 5
34
20
10
40
50
sorted array
10 20 34 40 50

...Program finished with exit code 0
Press ENTER to exit console.
```

ii) Insertion sort

## Program:

```
#include<stdio.h>

void insertionsort(int arr[],int n)
{
    int temp,i,j;
    for(int i=1;i<n;i++)
    {
        temp=arr[i];
        j=i-1;
        while(j>=0 && arr[j]>temp)
        {
            arr[j+1]=arr[j];
            j=j-1;
        }
        arr[j+1]=temp;
    }
}

int main()
{
    int n,a[10],i;

    printf("Enter the number of terms to sort ");
    scanf("%d",&n);
```

```
        for(int i=0;i<n;i++)
        {
            scanf("%d",&a[i]);
        }

        insertionsort(a,n);

        printf("sorted array\n");
        for(int i=0;i<n;i++)
        {
            printf("%d ",a[i]);
        }

        return 0;
    }
```

Output :

```
Enter the number of terms to sort 5
13
32
26
35
10
sorted array
10 13 26 32 35

...Program finished with exit code 0
Press ENTER to exit console.□
```

### iii)selection sort

#### Program:

```
#include<stdio.h>

void selectionsort(int arr[],int n)
{
    int temp,i,j,min;
    for(int i=0;i<n-1;i++)
    {
        min=i;
        for(j=i+1;j<n;j++)
        {
```

```
        if(arr[j]<arr[min])
        {
            min=j;
        }
        if(min!=i)
        {
            temp=arr[i];
            arr[i]=arr[min];
            arr[min]=temp;
        }
    }
}

int main()
{
    int n,a[10],i;

    printf("Enter the number of terms to sort ");
    scanf("%d",&n);

    for(int i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
}
```

```
selectionsort(a,n);
```

```
printf("sorted array\n");
```

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

```
    {
```

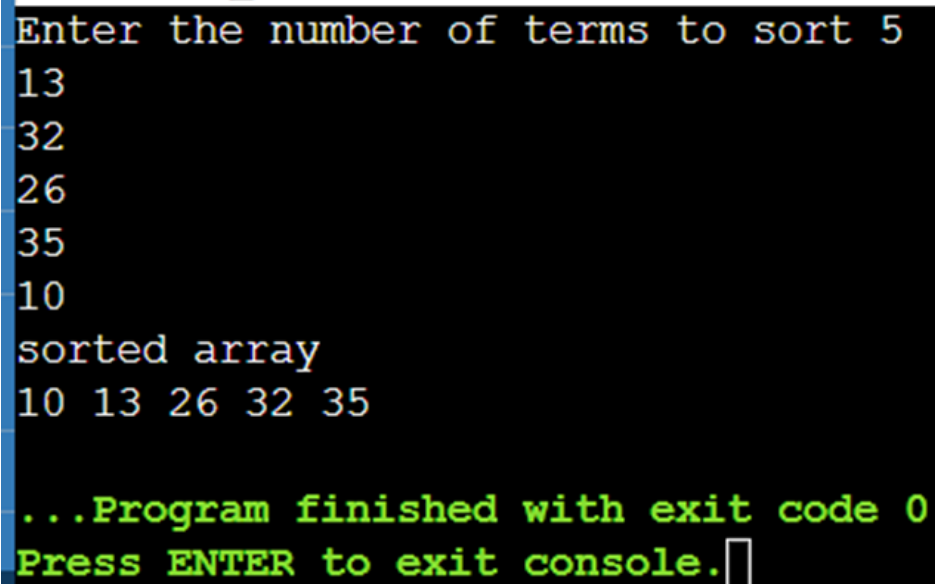
```
        printf("%d ",a[i]);
```

```
    }
```

```
    return 0;
```

```
}
```

Output :



```
Enter the number of terms to sort 5
13
32
26
35
10
sorted array
10 13 26 32 35

...Program finished with exit code 0
Press ENTER to exit console.
```

## WEEK - 10

1) Program :

```
// linear Search With No recursion

#include <stdio.h>

int main()
{
    int array[100], search, c, n;

    printf("Enter number of elements in array\n");
    scanf("%d", &n);

    printf("Enter %d integer(s)\n", n);

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

    printf("Enter a number to search\n");
    scanf("%d", &search);

    for (c = 0; c < n; c++)
    {
```



```
        if (array[c] == search) /* If required element is found */
        {
            printf("%d is present at location %d.\n", search, c+1);
            break;
        }
    }

    if (c == n)
        printf("%d isn't present in the array.\n", search);

    return 0;
}
```

Output :

```
Enter number of elements in array
5
Enter 5 integer(s)
23
46
58
68
34
Enter a number to search
68
68 is present at location 4.

...Program finished with exit code 0
Press ENTER to exit console.□
```

2) Program :

```
// Linear Search with Recursion

#include <stdio.h>

int RecursiveLS(int arr[], int value, int index, int n)
{
    int pos = 0;

    if(index >= n)
    {
```

```
return 0;
```

```
}
```

```
else if (arr[index] == value)
```

```
{
```

```
pos = index + 1;
```

```
return pos;
```

```
}
```

```
else
```

```
{
```

```
return RecursiveLS(arr, value, index+1, n);
```

```
}
```

```
return pos;
```

```
}
```

```
int main()
```

```
{
```

```
int n, value, pos, m = 0, arr[100];
```

```
printf("Enter the total elements in the array ");
```

```
scanf("%d", &n);
```

```
printf("Enter the array elements\n");
```

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

```
    {
```

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

```
    }
```

```
printf("Enter the element to search ");
```

```
scanf("%d", &value);
```

```
    pos = RecursiveLS(arr, value, 0, n);
```

```
    if (pos != 0)
```

```
    {
```

```
        printf("Element found at pos %d ", pos);
```

```
    }
```

```
    else
```

```
    {
```

```
        printf("Element not found");
```

```
    }
```

```
    return 0;
```

```
}
```

Output :

```
Enter the total elements in the array 5
Enter the array elements
1
2
4
5
8
Enter the element to search 5
Element found at pos 4

...Program finished with exit code 0
Press ENTER to exit console.□
```

3) Program :

```
// Binary Search with out recursion

#include <stdio.h>

int main()
{
    int c, first, last, middle, n, search, array[100];

    printf("Enter number of elements\n");
    scanf("%d", &n);

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

```
for (c = 0; c < n; c++)
```

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

```
printf("Enter value to find\n");
```

```
scanf("%d", &search);
```

```
first = 0;
```

```
last = n - 1;
```

```
middle = (first+last)/2;
```

```
while (first <= last) {
```

```
    if (array[middle] < search)
```

```
        first = middle + 1;
```

```
    else if (array[middle] == search) {
```

```
        printf("%d found at location %d.\n", search, 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", search);  
  
return 0;  
  
}
```

Output :

```
Enter number of elements  
5  
Enter 5 integers  
1  
2  
3  
4  
5  
Enter value to find  
4  
4 found at location 4.  
  
...Program finished with exit code 0  
Press ENTER to exit console. 
```

4) Program :

```
// binary search with recursion
```

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

```
int recursiveBinarySearch(int array[], int start_index, int end_index, int  
element){
```

```
    if (end_index >= start_index){
```

```
        int middle = start_index + (end_index - start_index )/2;
```

```
        if (array[middle] == element)
```

```
            return middle;
```

```
        if (array[middle] > element)
```

```
            return recursiveBinarySearch(array, start_index, middle-1, element);
```

```
            return recursiveBinarySearch(array, middle+1, end_index, element);
```

```
    }
```

```
    return -1;
```

```
}
```

```
int main(void){
```

```
    int array[10];
```

```
    int n,i;
```

```
    printf("Enter the number of elements in array to search\n");
```

```
    scanf("%d",&n);
```

```
    printf("Enter %d elements",n);
```

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

```
    {
```

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



```

}

int element;

printf("Enter the element to search\n");

scanf("%d",&element);

int found_index = recursiveBinarySearch(array, 0, n-1, element);

if(found_index == -1 ) {

    printf("Element not found in the array ");

}

else {

    printf("Element found at index : %d",found_index+1);

}

return 0;

}

```

Output :

```

Enter the number of elements in array to search
5
Enter 5 elements1
2
3
4
5
Enter the element to search
4
Element found at index : 4

...Program finished with exit code 0
Press ENTER to exit console.

```

