

Laboratory Manual
of
Programming for Problem Solving



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ANURAG COLLEGE OF ENGINEERING

Aushapur (V), Ghatkesar (M), Medchal Dist.

CS106ES/CS206ES: PROGRAMMING FOR PROBLEM SOLVING LAB

B.Tech. I Year I Sem.

L T P C
0 0 3 1.5

[Note: The programs may be executed using any available Open Source/ Freely available IDE

Some of the Tools available are:

CodeLite: <https://codelite.org/>

Code::Blocks: <http://www.codeblocks.org/>

DevCpp : <http://www.bloodshed.net/devcpp.html>

Eclipse: <http://www.eclipse.org>

This list is not exhaustive and is NOT in any order of preference]

Course Objectives: The students will learn the following:

- To work with an IDE to create, edit, compile, run and debug programs
- To analyze the various steps in program development.
- To develop programs to solve basic problems by understanding basic concepts in C like operators, control statements etc.
- To develop modular, reusable and readable C Programs using the concepts like functions, arrays etc.
- To Write programs using the Dynamic Memory Allocation concept.
- To create, read from and write to text and binary files

Course Outcomes: The candidate is expected to be able to:

- formulate the algorithms for simple problems
- translate given algorithms to a working and correct program
- correct syntax errors as reported by the compilers
- identify and correct logical errors encountered during execution
- represent and manipulate data with arrays, strings and structures
- use pointers of different types
- create, read and write to and from simple text and binary files
- modularize the code with functions so that they can be reused

Practice sessions:

- a. Write a simple program that prints the results of all the operators available in C (including pre/ post increment , bitwise and/or/not , etc.). Read required operand values from standard input.
- b. Write a simple program that converts one given data type to another using auto conversion and casting. Take the values form standard input.

Simple numeric problems:

- a. Write a program for fiend the max and min from the three numbers.
- b. Write the program for the simple, compound interest.

- c. Write program that declares Class awarded for a given percentage of marks, where mark <40%= Failed, 40% to <60% = Second class, 60% to <70%=First class, >= 70% = Distinction. Read percentage from standard input.
- d. Write a program that prints a multiplication table for a given number and the number of rows in the table. For example, for a number 5 and rows = 3, the output should be:
- e. $5 \times 1 = 5$
- f. $5 \times 2 = 10$
- g. $5 \times 3 = 15$
- h. Write a program that shows the binary equivalent of a given positive number between 0 to 255.

Expression Evaluation:

- a. A building has 10 floors with a floor height of 3 meters each. A ball is dropped from the top of the building. Find the time taken by the ball to reach each floor. (Use the formula $s = ut + \frac{1}{2}at^2$ where u and a are the initial velocity in m/sec (= 0) and acceleration in m/sec^2 (= $9.8 m/s^2$)).
- b. 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)
- c. Write a program that finds if a given number is a prime number
- d. Write a C program to find the sum of individual digits of a positive integer and test given number is palindrome.
- e. A 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.
- f. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.
- g. Write a C program to find the roots of a Quadratic equation.
- h. Write a C program to calculate the following, where x is a fractional value.
- i. $1 - \frac{x}{2} + \frac{x^2}{4} - \frac{x^3}{6}$
- j. 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 $1 + 5 + 25 + 125$.

Arrays and Pointers and Functions:

- a. Write a C program to find the minimum, maximum and average in an array of integers.
- b. Write a functions to compute mean, variance, Standard Deviation, sorting of n elements in single dimension array.
- c. Write a C program that uses functions to perform the following:
- d. Addition of Two Matrices
- e. ii. Multiplication of Two Matrices

- f. iii. Transpose of a matrix with memory dynamically allocated for the new matrix as row and column counts may not be same.
- g. Write C programs that use both recursive and non-recursive functions
- h. To find the factorial of a given integer.
- i. ii. To find the GCD (greatest common divisor) of two given integers.
- j. iii. To find x^n
- k. Write a program for reading elements using pointer into array and display the values using array.
- l. Write a program for display values reverse order from array using pointer.
- m. Write a program through pointer variable to sum of n elements from array.

Files:

- a. Write a C program to display the contents of a file to standard output device.
- b. Write a C program which copies one file to another, replacing all lowercase characters with their uppercase equivalents.
- c. Write a C program to count the number of times a character occurs in a text file. The file name and the character are supplied as command line arguments.
- d. Write a C program that does the following:
It should first create a binary file and store 10 integers, where the file name and 10 values are given in the command line. (hint: convert the strings using atoi function)
Now the program asks for an index and a value from the user and the value at that index should be changed to the new value in the file. (hint: use fseek function)
The program should then read all 10 values and print them back.
- e. 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).

Strings:

- a. Write a C program to convert a Roman numeral ranging from I to L to its decimal equivalent.
- b. Write a C program that converts a number ranging from 1 to 50 to Roman equivalent
- c. Write a C program that uses functions to perform the following operations:
- d. To insert a sub-string in to a given main string from a given position.
- e. ii. To delete n Characters from a given position in a given string.
- f. Write a C program to determine if the given string is a palindrome or not (Spelled same in both directions with or without a meaning like madam, civic, noon, abcba, etc.)
- g. Write a C program that displays the position of a character ch in the string S or – 1 if S doesn't contain ch.
- h. Write a C program to count the lines, words and characters in a given text.

Miscellaneous:

- a. Write a menu driven C program that allows a user to enter n numbers and then choose between finding the smallest, largest, sum, or average. The menu and all the choices

are to be functions. Use a switch statement to determine what action to take. Display an error message if an invalid choice is entered.

b. Write a C program to construct a pyramid of numbers as follows:

```
1           *           1           1           *
1 2         * *         2 3         2 2         * *
1 2 3       * * *       4 5 6       3 3 3       * * *
                                         4 4 4 4       * *
                                         *
                                         *
```

Sorting and Searching:

- a. Write a C program that uses non recursive function to search for a Key value in a given
- b. list of integers using linear search method.
- c. Write a C program that uses non recursive function to search for a Key value in a given
- d. sorted list of integers using binary search method.
- e. Write a C program that implements the Bubble sort method to sort a given list of
- f. integers in ascending order.
- g. Write a C program that sorts the given array of integers using selection sort in descending order
- h. Write a C program that sorts the given array of integers using insertion sort in ascending order
- i. Write a C program that sorts a given array of names

Suggested Reference Books for solving the problems:

- i. Byron Gottfried, Schaum's Outline of Programming with C, McGraw-Hill
- ii. B.A. Forouzan and R.F. Gilberg C Programming and Data Structures, Cengage Learning, (3rd Edition)
- iii. Brian W. Kernighan and Dennis M. Ritchie, The C Programming Language, Prentice
- iv. Hall of India
- v. R.G. Dromey, How to solve it by Computer, Pearson (16th Impression)
- vi. Programming in C, Stephen G. Kochan, Fourth Edition, Pearson Education.
- vii. Herbert Schildt, C: The Complete Reference, Mc Graw Hill, 4th Edition

PREFACE

This manual was developed specifically for fresh students taking up their first course in programming. Its aim is to supplement classroom lectures by focusing on C programming. Topics are arranged based on the order of class discussion. Students will be exposed to several new aspects of programming. Students will get knowledge about the concepts in C programming which includes functions, arrays, pointers, structures, unions, e.t.c.

Course Objectives: The students will learn the following:

- To work with an IDE to create, edit, compile, run and debug programs
- To analyze the various steps in program development.
- To develop programs to solve basic problems by understanding basic concepts in C like operators, control statements etc.
- To develop modular, reusable and readable C Programs using the concepts like functions, arrays etc.
- To write programs using the Dynamic Memory Allocation concept.
- To create, read from and write to text and binary files.

Course Outcomes: The candidate is expected to be able to:

- formulate the algorithms for simple problems
- translate given algorithms to a working and correct program
- correct syntax errors as reported by the compilers
- identify and correct logical errors encountered during execution
- represent and manipulate data with arrays, strings and structures
- use pointers of different types
- create, read and write to and from simple text and binary files
- Modularize the code with functions so that they can be reused.

Some of the Tools available are:

- CodeLite: <https://codelite.org/>
- Code::Blocks: <http://www.codeblocks.org/>
- DevCpp : <http://www.bloodshed.net/devcpp.html>
- Eclipse: <http://www.eclipse.org>
- This list is not exhaustive and is NOT in any order of preference]

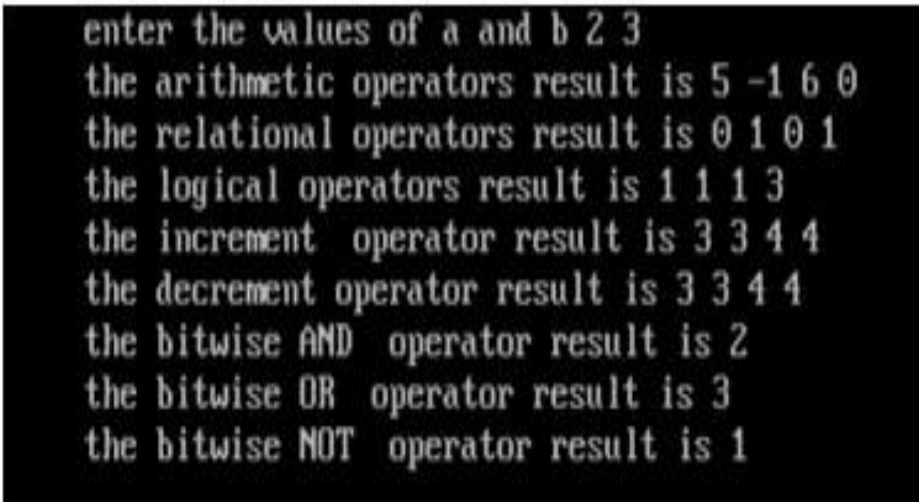
Practice sessions:

- a. Write a simple program that prints the results of all the operators available in C (including pre/post increment, bitwise and/or/not, etc.). Read required operand values from standard input.

Program:

```
#include<stdio.h>
Void main( )
{
int a,b;
printf("enter the values of a and b");
scanf("%d%d",&a,&b);
printf("the arithmetic operators result is %d %d %d %d", a+b,a-b,a*b,a/b);
printf("the relational operators result is %d %d %d %d", a>b,a<b,a>=b,a<=b);
printf("the logical operators result is %d %d %d %d", a&&b,a||b,!(a==b));
printf("the increment operator result is %d %d %d %d",a++,++a,b++,++b);
printf("the decrement operator result is %d %d %d %d",a--,--a,b--,--b);
printf("the bitwise AND operator result is %d",a&b);
printf("the bitwise OR operator result is %d",a|b);
printf("the bitwise NOT operator result is %d",a^b);
}
```

Output:

A screenshot of a terminal window showing the output of the C program. The text is as follows:

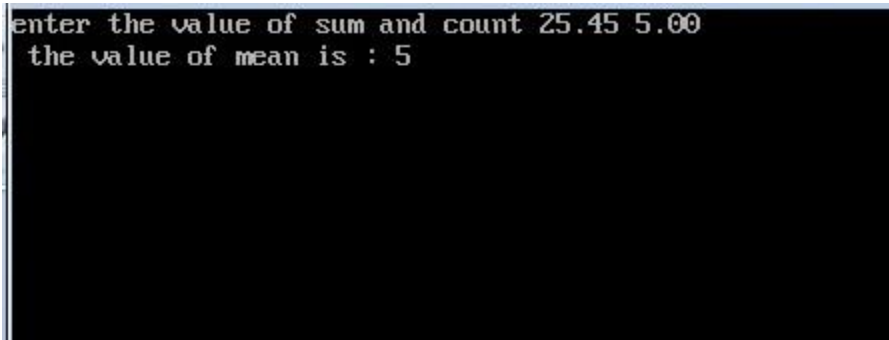
```
enter the values of a and b 2 3
the arithmetic operators result is 5 -1 6 0
the relational operators result is 0 1 0 1
the logical operators result is 1 1 1 3
the increment operator result is 3 3 4 4
the decrement operator result is 3 3 4 4
the bitwise AND operator result is 2
the bitwise OR operator result is 3
the bitwise NOT operator result is 1
```

- b. Write a simple program that converts one given data type to another using auto conversion and casting. Take the values form standard input.

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
float sum,count;
int mean;
clrscr();
printf("enter the value of sum and count");
scanf("%f%f",&sum,&count);
mean=sum/count;
printf(" the value of mean is : %d\n", mean);
}
```

Output:

A screenshot of a terminal window with a black background and white text. The text shows the program's execution: "enter the value of sum and count 25.45 5.00" followed by "the value of mean is : 5".

```
enter the value of sum and count 25.45 5.00
the value of mean is : 5
```

Simple numeric problems:

a. Write a program for fiend the max and min from the three numbers.

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a,b,c;
    clrscr();
    printf("Enter 3 numbers");
    scanf("%d%d%d",&a,&b,&c);
    if(a>b && a>c)
        printf("Maximum number is a = %d",a);
    else if(b>a && b>c)
        printf("Maximum number is b = %d",b);
    else
        printf("Maximum number is c = %d",c);
    printf("\n");
    if(a<b && a<c)
        printf("Minimum number is a = %d",a);
    else if(b<a && b<c)
        printf("Minimum number is b = %d",b);
    else
        printf("Minimum number is c = %d",c);
    getch();
}
```

Output:

```
Enter 3 numbers 5 6 3
Maximum number is b = 6
Minimum number is c = 3
```

b. Write the program for the simple, compound interest.

Program:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
    int p,t;
    float r,si,amount,ci;
    clrscr( );
    printf("Please enter principle,time and rate of interest");
    scanf("%d%d%f",&p,&t,&r);
    si=p*t*r/100;
    printf("\nSimple interest = %.3f",si);
    amount=p*pow((1 +r/100),t);
    ci=amount-p;
    printf("\nCompound interest = %.3f",ci);
    getch( );
}
```

Output:

```
Please enter principle,time and rate of interest 500 12 1
Simple interest = 60.000
Compound interest = 63.413
```

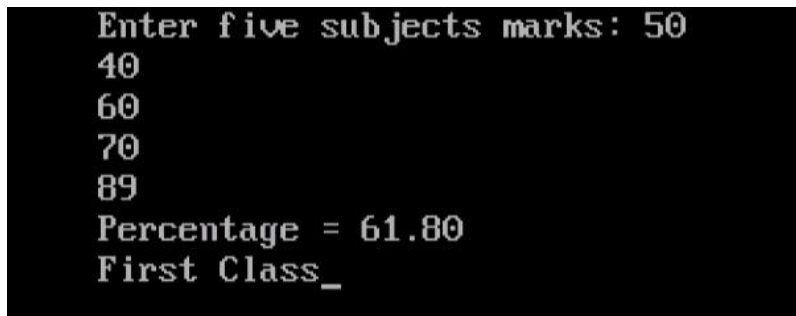
- c. Write program that declares Class awarded for a given percentage of marks, where mark = 70% = Distinction. Read percentage from standard input.

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int phy, chem, bio, math, comp;
    float per;
    clrscr();
    printf("Enter five subjects marks: ");
    scanf("%d%d%d%d%d", &phy, &chem, &bio, &math, &comp);
    per = (phy + chem + bio + math + comp) / 5.0;
    printf("Percentage = %.2f\n", per);
    if(per >= 70)
    {
        printf("Distinction");
    }
    else if(per <=40 )
    {
        printf("Failed");
    }
    else if(per <= 60 && per>40)
    {
        printf("Second Class");
    }
    else if(per <= 70 && per>60)
    {
        printf("First Class");
    }
}

getch();
}
```

Output:



```
Enter five subjects marks: 50
40
60
70
89
Percentage = 61.80
First Class_
```

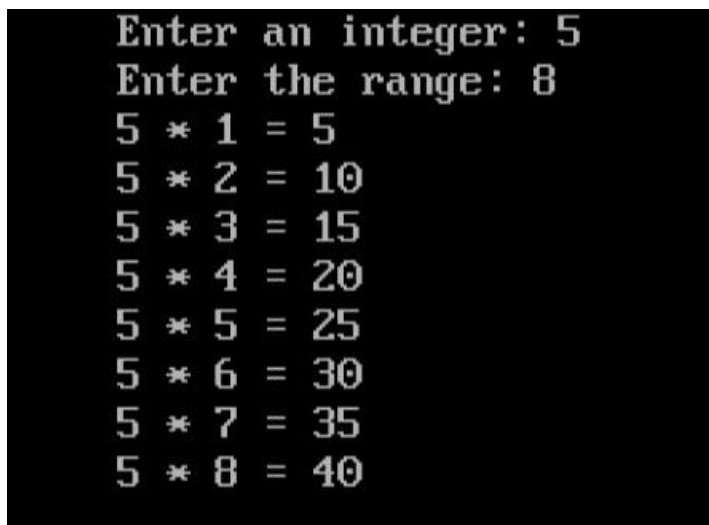
d. Write a program that prints a multiplication table for a given number and the number of rows in the table. For example, for a number 5 and rows = 3, the output should be:

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15

Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int n, i, range;
    clrscr();
    printf("Enter an integer: ");
    scanf("%d",&n);
    printf("Enter the range: ");
    scanf("%d", &range);
    for(i=1; i<=range;i++)
    {
        printf("%d * %d = %d \n",n,i, n*i);
    }
    getch();
}
```

Output:



```
Enter an integer: 5
Enter the range: 8
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
```

e. Write a program that shows the binary equivalent of a given positive number between 0 to 255.

Program:

```
#include <stdio.h>
#include <conio.h>
int binary_conversion(int);
void main()
{
    int num, bin;
    clrscr();
    printf("Enter a decimal number: ");
    scanf("%d", &num);
    bin = binary_conversion(num);
    printf("The binary equivalent of %d is %d\n", num, bin);
    getch();
}
int binary_conversion(int num)
{
    if (num == 0)
    {
        return 0;
    }
    else
    {
        return (num % 2) + 10 * binary_conversion(num / 2);
    }
}
```

Output:

```
Enter a decimal number: 10
The binary equivalent of 10 is 1010
```

Expression Evaluation:

a. A building has 10 floors with a floor height of 3 meters each. A ball is dropped from the top of the building. Find the time taken by the ball to reach each floor. (Use the formula $s = ut + \frac{1}{2}at^2$ where u and a are the initial velocity in m/sec (= 0) and acceleration in m/sec² (= 9.8 m/s²)).

b. 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).

Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a, b, c;
    char ch;
    clrscr() ;
    printf("Enter your operator(+, -, /, *, %)\n");
    scanf("%c", &ch);
    printf("Enter the values of a and b\n");
    scanf("%d%d", &a, &b);
    switch(ch)
    {
        case '+':    c = a + b;
                    printf("addition of two numbers is %d", c);
                    break;
        case '-':    c = a - b;
                    printf("substraction of two numbers is %d", c);
                    break;
        case '*':    c = a * b;
                    printf("multiplication of two numbers is %d", c);
                    break;
        case '/':    c = a / b;
                    printf("remainder of two numbers is %d", c);
                    break;
        case '%':    c = a % b;
                    printf("quotient of two numbers is %d", c);
                    break;
        default:    printf("Invalid operator");
                    break;
    }
    getch();
}
```

Output:

```
Enter your operator(+, -, /, *, %)
+
Enter the values of a and b
4 5
addition of two numbers is 9
```

c. Write a program that finds if a given number is a prime number

Program:

```
#include<stdio.h>
int main()
{
    int n,i,flag=0;
    printf("\nEnter a number:");
    scanf("%d",&n);
    for(i=2;i<=n/2;i++)
    {
        if(n%i==0)
        { flag=1; break;
        }
    }
    If(flag==0)
        printf("%d is a prime number",n);
    else
        printf("%d is not a prime number",n);
    return(0);
}
```

Output:

Enter a number: 29

29 is a prime number

d. Write a C program to find the sum of individual digits of a positive integer and test given number is palindrome.

Sum of individual digits of a positive integer:

Program:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main ()
{
int number = 0, digit = 0, sum = 0;
clrscr();
printf("Enter any number\n ");
scanf("%d", &number);
while (number != 0)
{
digit = number % 10;
sum = sum + digit;
number = number / 10;
}
printf ("Sum of individual digits of a given number is %d", sum);
getch();
}
```

Output:

```
Enter any number
105
Sum of individual digits of a given number is 6_
```

Test given number is palindrome.

Program:

```
#include <stdio.h>
void main()
{
    int number, t, rev=0, rmndr;
    printf("Please enter a number to check Palindrome : ");
    scanf("%d",&number);
    printf("\nEntered number: %d", number);
    t = number;
    while (number > 0)
    {
        rmndr = number%10;
        rev = rev*10 + rmndr;
        number = number/10;
    }
    printf("\nReversed number: %d", rev);
    if(t == rev)
    {
        printf("\nEntered number %d is a palindrome", t);
    }
    else
    {
        printf("\nEntered number %d is not a palindrome", t);
    }
}
```

Output:

```
Please enter a number to check Palindrome : 141

Entered number: 141
Reversed number: 141
Entered number 141 is a palindrome_
```

e) A 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.

Program:

```
#include <stdio.h>
int main()
{
    int i, n, t1 = 0, t2 = 1, nextTerm;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    printf("Fibonacci Series: ");
    for (i = 1; i <= n; ++i)
    {
        printf("%d, ", t1);
        nextTerm = t1 + t2;
        t1 = t2;
        t2 = nextTerm;
    }
    return 0;
}
```

Output

```
Enter the number of terms: 10
Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34,
```

f. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.

Program:

```
#include<stdio.h>
void main()
{
int m,n,i,nof;
printf("\nEnter a number:");
scanf("%d",&n);
printf("\nThe Prime numbers between 1 to n:");
for(m=2;m<=n;m++)
    {
    nof=0;
        for(i=2;i<=m/2;i++)
            {
                if(m%i==0)
                    nof++;
            }
        if(nof==0)
            printf("%d",m);
    }
}
```

OUTPUT:

Enter the value of n: 40

The Prime numbers between 1 to n: 2 3 5 7 11 13 17

19 23 29 31 37

g. Write a C program to find the roots of a Quadratic equation.

Program:

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

    int a,b,c,disc;
    float root1,root2;
    clrscr();
    printf("ENTER VALUES FOR a,b,c:\n");
    scanf("%d %d %d",&a,&b,&c);
    disc=b*b-4*a*c;
    if(disc>0)
    {

        printf("THE ROOTS ARE REAL & THEY ARE..\n");
        root1=(-b+sqrt(disc))/(2*a);
        root2=(-b-sqrt(disc))/(2*a);
        printf("Root1=%f",root1);
        printf("Root2=%f",root2);

    }

    else if(disc==0)
    {
        printf("THE ROOTS ARE EQUAL & THEY ARE..\n");
        root1=-b/(2*a);
        root2=root1;
    }
}
```

```
        printf("Root1=%f",root1);
        printf("Root2=%f",root2);
    }
    else
    {
        printf("THE ROOTS ARE IMAGINARY");
        x=-b/(2*a);
        y= sqrt(disc)/(2*a);
        printf("Root1=%f+i%f",x,y);
        printf("Root1=%f-i%f",x,y);
    }

    getch();
}
```

OUTPUT:

ENTER VALUES FOR a,b,c 1 4 4
THE ROOTS ARE EQUAL AND THEY ARE.. Root1=-2 Root2=-2

h. Write a C program to calculate the following, where x is a fractional value.

$$1 - \frac{x}{2} + \frac{x^2}{4} - \frac{x^3}{6}$$

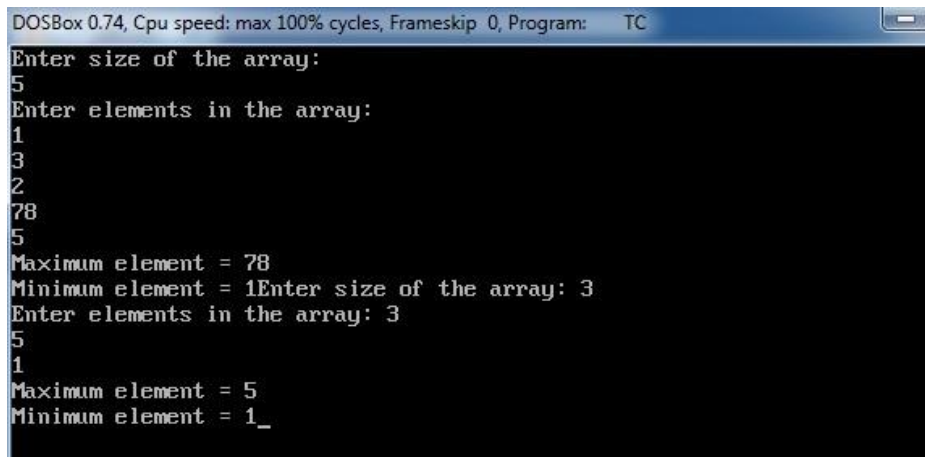
j. Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression: $1+x+x^2+x^3+\dots+x^n$. For example: if n is 3 and x is 5, then the program computes $1+5+25+125$.

Arrays and Pointers and Functions:

- f. Write a C program to find the minimum, maximum and average in an array of integers.

```
#include <stdio.h>
#define MAX_SIZE 100 // Maximum array size
int main()
{
    int arr[MAX_SIZE];
    int i, max, min, size;
    printf("Enter size of the array: ");
    scanf("%d", &size);
    printf("Enter elements in the array: ");
    for(i=0; i<size; i++)
    {
        scanf("%d", &arr[i]);
    }
    max = arr[0];
    min = arr[0];
    for(i=1; i<size; i++)
    {
        if(arr[i] > max)
        {
            max = arr[i];
        }
        if(arr[i] < min)
        {
            min = arr[i];
        }
    }
    printf("Maximum element = %d\n", max);
    printf("Minimum element = %d", min);
    return 0;
}
```

Output:



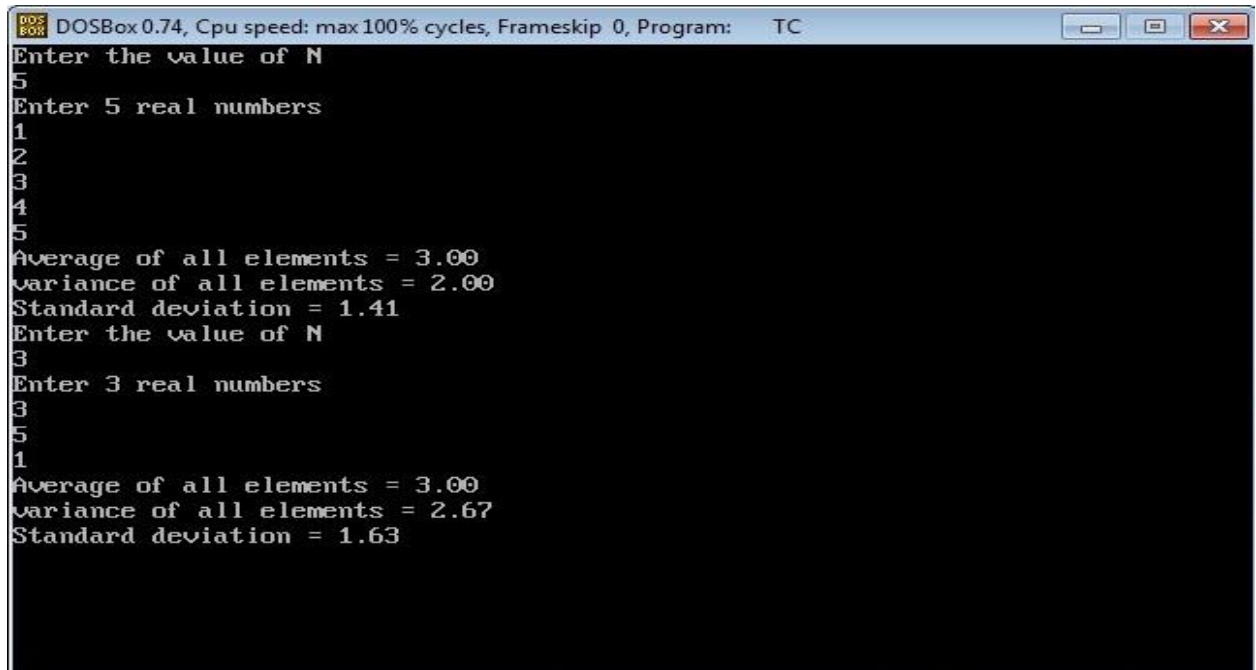
```
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter size of the array:
5
Enter elements in the array:
1
3
2
78
5
Maximum element = 78
Minimum element = 1Enter size of the array: 3
Enter elements in the array: 3
5
1
Maximum element = 5
Minimum element = 1_
```

2. Write a functions to compute mean, variance, Standard Deviation, sorting of n elements in single dimension array.

Program:

```
#include <stdio.h>
#include <math.h>
#define MAXSIZE 10
void main()
{
    float x[MAXSIZE];
    int i, n;
    float average, variance, std_deviation, sum = 0, sum1 = 0;
    printf("Enter the value of N \n");
    scanf("%d", &n);
    printf("Enter %d real numbers \n", n);
    for (i = 0; i < n; i++)
    {
        scanf("%f", &x[i]);
    }
    /* Compute the sum of all elements */
    for (i = 0; i < n; i++)
    {
        sum = sum + x[i];
    }
    average = sum / (float)n;
    for (i = 0; i < n; i++)
    {
        sum1 = sum1 + pow((x[i] - average), 2);
    }
    variance = sum1 / (float)n;
    std_deviation = sqrt(variance);
    printf("Average of all elements = %.2f\n", average);
    printf("variance of all elements = %.2f\n", variance);
    printf("Standard deviation = %.2f\n", std_deviation);
    getch();
}
```

Output:



```
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter the value of N
5
Enter 5 real numbers
1
2
3
4
5
Average of all elements = 3.00
variance of all elements = 2.00
Standard deviation = 1.41
Enter the value of N
3
Enter 3 real numbers
3
5
1
Average of all elements = 3.00
variance of all elements = 2.67
Standard deviation = 1.63
```

3 Write a C program that uses functions to perform the following:
Addition of Two Matrices

```
#include <stdio.h>

int main()
{
    int m, n, c, d, first[10][10], second[10][10], sum[10][10];
    clrscr();

    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");

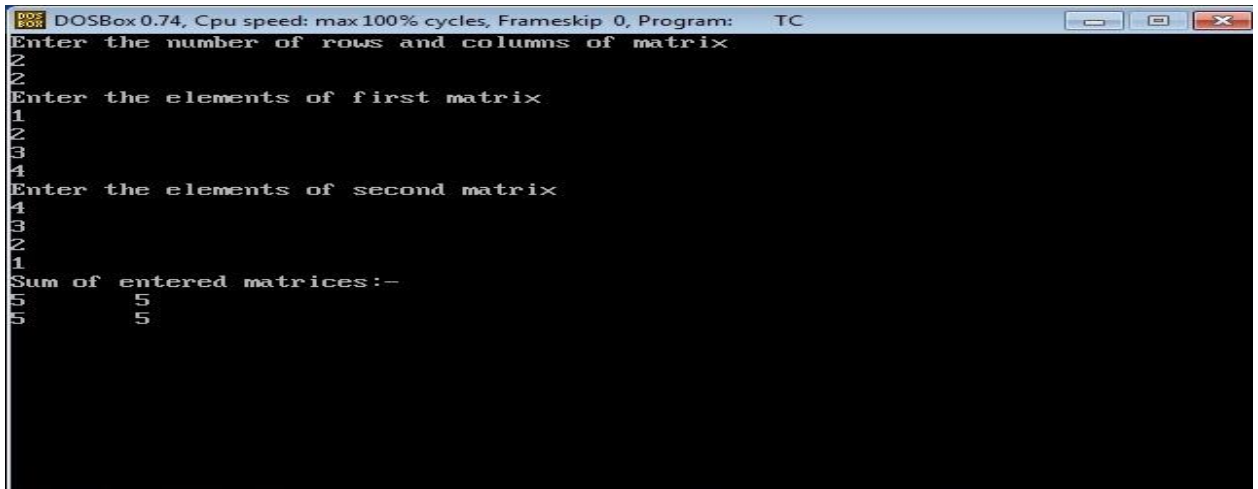
    for (c = 0; c < m; c++)
        for (d = 0 ; d < n; d++)
            scanf("%d", &second[c][d]);

    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");
    }

    return 0;
}
```

Output:



```
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter the number of rows and columns of matrix
2
2
Enter the elements of first matrix
1
1
3
4
Enter the elements of second matrix
4
3
1
Sum of entered matrices:-
5
5
```

Multiplication of Two Matrices

Program:

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



```

        for (k = 0; k < p; k++) {
            sum = sum + first[c][k]*second[k][d];
        }
        multiply[c][d] = sum;
        sum = 0;
    }
}
printf("Product of the matrices:\n");
for (c = 0; c < m; c++) {
    for (d = 0; d < q; d++)
        printf("%d\t", multiply[c][d]);
    printf("\n");
}
}
getch();
return 0;
}

```

Output:

```

DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter number of rows and columns of first matrix
2 2
Enter elements of first matrix
1 2 3 4
Enter number of rows and columns of second matrix
2 2
Enter elements of second matrix
4
3
2
1
Product of the matrices:
8      5
20     13

```

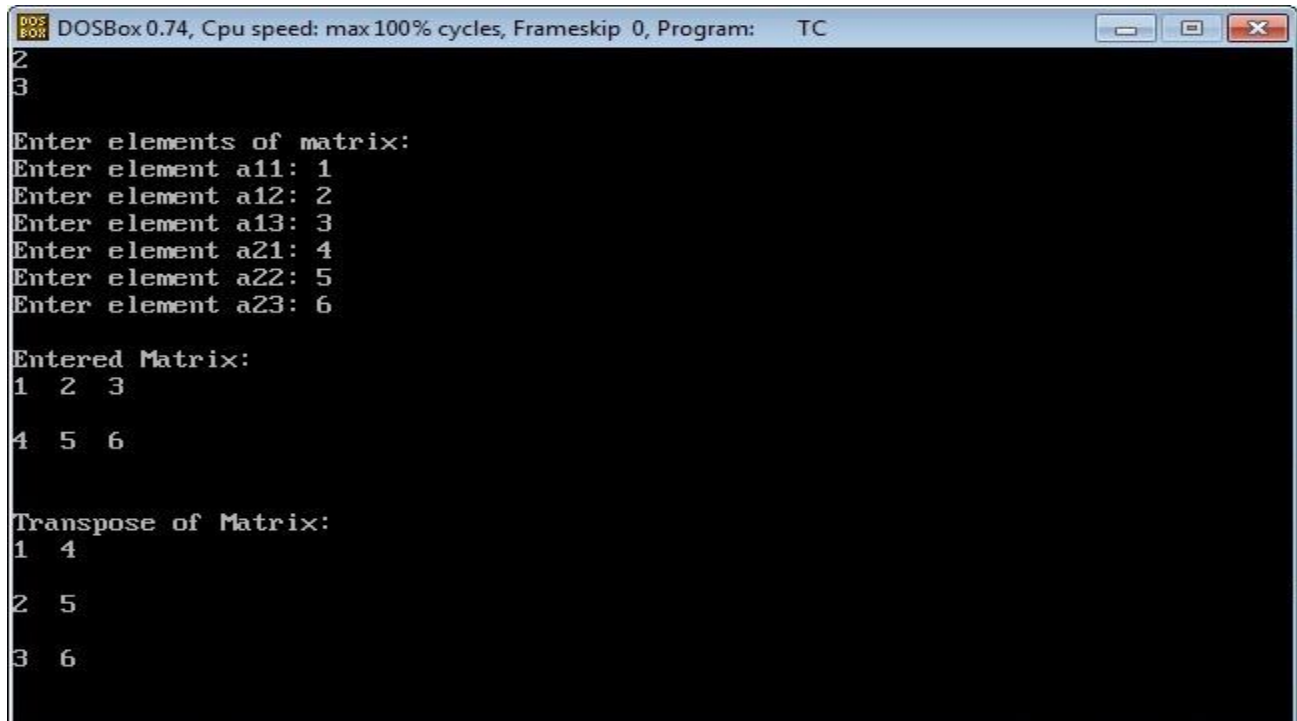
Transpose of a matrix with memory dynamically allocated for the new matrix as row and column counts may not be same

Program:

```
#include <stdio.h>
int main()
{
    int a[10][10], transpose[10][10], r, c, i, j;
    printf("Enter rows and columns of matrix: ");
    scanf("%d %d", &r, &c);
    // Storing elements of the matrix
    printf("\nEnter elements of matrix:\n");
    for(i=0; i<r; ++i)
        for(j=0; j<c; ++j)
        {
            printf("Enter element a%d%d: ", i+1, j+1);
            scanf("%d", &a[i][j]);
        }
    // Displaying the matrix a[][] */
    printf("\nEnter Matrix: \n");
    for(i=0; i<r; ++i)
        for(j=0; j<c; ++j)
        {
            printf("%d ", a[i][j]);
            if (j == c-1)
                printf("\n\n");
        }
    // Finding the transpose of matrix a
    for(i=0; i<r; ++i)
        for(j=0; j<c; ++j)
        {
            transpose[j][i] = a[i][j];
        }
    // Displaying the transpose of matrix a
    printf("\nTranspose of Matrix:\n");
    for(i=0; i<c; ++i)
        for(j=0; j<r; ++j)
        {
            printf("%d ", transpose[i][j]);
        }
}
```

```
        if(j==r-1)
            printf("\n\n");
    }
    return 0;
}
```

Output:



```
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
2
3
Enter elements of matrix:
Enter element a11: 1
Enter element a12: 2
Enter element a13: 3
Enter element a21: 4
Enter element a22: 5
Enter element a23: 6
Entered Matrix:
1 2 3
4 5 6
Transpose of Matrix:
1 4
2 5
3 6
```

Write C programs that use both recursive and non-recursive functions To compute x^n using recursive.

Program:

```
#include<stdio.h>
int power(int n1,int n2);
int main()
{
    int base,exp;
    printf("enter base number:");
    scanf("%d",&base);
    printf("enter power number(positive integer):");
    scanf("%d",&exp);
    printf("%d^%d=%d",base,exp,power(base,exp));
    return 0;
}

int power(int base,int exp)
{
    if(exp!=1)
        return (base*power(base,exp-1));
}
```

OUTPUT:

```
enter base number:3
enter power number(positive integer):3
3^3=27
```

Write C programs that use both recursive and non-recursive functions

- i. To find the factorial of a given integer.

Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int n, a, b;
    clrscr();
    printf("Enter any number\n");
    scanf("%d", &n);
    a = recfactorial(n);
    printf("The factorial of a given number using recursion is %d \n", a);
    b = nonrecfactorial(n);
    printf("The factorial of a given number using nonrecursion is %d ", b);
    getch();
}
int recfactorial(int x)
{
    int f;
    if(x == 0)
    {
        return(1);
    }
    else
    {
        f = x * recfactorial(x - 1);
        return(f);
    }
}
int nonrecfactorial(int x)
{
    int i, f = 1;
    for(i = 1; i <= x; i++)
    {
        f = f * i;
    }
}
```

```
    return(f);  
}
```

Output:

Enter any number

5

The factorial of a given number using recursion is 120

The factorial of a given number using nonrecursion is 120

ii. To find the GCD (greatest common divisor) of two given integers.

Program:

```
#include <stdio.h>  
#include <conio.h>  
void main()  
{  
    int a, b, c, d;  
    clrscr();  
    printf("Enter two numbers a, b\n");  
    scanf("%d%d", &a, &b);  
    c = recgcd(a, b);  
    printf("The gcd of two numbers using recursion is %d\n", c);  
    d = nonrecgcd(a, b);  
    printf("The gcd of two numbers using nonrecursion is %d", d);  
    getch();  
}  
int recgcd(int x, int y)  
{  
    if(y == 0)  
    {  
        return(x);  
    }  
    else  
    {  
        return(recgcd(y, x % y));  
    }  
}
```

```
}  
int nonrecgcd(int x, int y)  
{  
    int z;  
    while(x % y != 0)  
    {  
        z = x % y;  
        x = y;  
        y = z;  
    }  
    return(y);  
}
```

Output:

Enter two numbers a, b

3 6

The gcd of two numbers using recursion is 3

The gcd of two numbers using nonrecursion is 3

- n. Write a program for reading elements using pointer into array and display the values using array.

Program:

```
#include <stdio.h>

int main()
{
    int data[5], i;
    printf("Enter elements: ");
    for(i = 0; i < 5; ++i)
        scanf("%d", data + i);
    printf("You entered: \n");
    for(i = 0; i < 5; ++i)
        printf("%d\n", *(data + i));
    return 0;
}
```

Output:

Enter elements:

1

2

3

5

4

You entered:

1

2

3

5

4

Write a C program using pointers to read in an array of integers and print its elements in reverse order.

Program:

```
#include<stdio.h>
#include<conio.h>
#define MAX 30

void main() {
    int size, i, arr[MAX];
    int *ptr;
    clrscr();

    ptr = &arr[0];

    printf("\nEnter the size of array : ");
    scanf("%d", &size);

    printf("\nEnter %d integers into array: ", size);
    for (i = 0; i < size; i++) {
        scanf("%d", ptr);
        ptr++;
    }

    ptr = &arr[size - 1];

    printf("\nElements of array in reverse order are :");

    for (i = size - 1; i >= 0; i--) {
        printf("\nElement%d is %d : ", i, *ptr);
        ptr--;
    }

    getch();
}
```

Output :

Enter the size of array : 5

Enter 5 integers into array : 11 22 33 44 55

Elements of array in reverse order are :

Element 4 is : 55

Element 4 is : 44

Element 4 is : 33

Element 4 is : 22

Element 4 is : 11

Write a 'C' Program to compute the sum of all elements stored in an array using pointers

Program:

```
#include<stdio.h>
#include<conio.h>
void main() {
    int numArray[10];
    int i, sum = 0;
    int *ptr;

    printf("\nEnter 10 elements : ");

    for (i = 0; i < 10; i++)
        scanf("%d", &numArray[i]);

    ptr = numArray; /* a=&a[0] */

    for (i = 0; i < 10; i++) {
        sum = sum + *ptr;
        ptr++;
    }
    printf("The sum of array elements : %d", sum);
}
```

Output:

Enter 10 elements: 11 12 13 14 15 16 17 18 19 20

The sum of array elements is 155

FILES

1. C Program to print contents of file

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    FILE *fptr;

    char filename[100], c;

    printf("Enter the filename to open \n");
    scanf("%s", filename);

    // Open file
    fptr = fopen(filename, "r");
    if (fptr == NULL)
    {
        printf("Cannot open file \n");
        exit(0);
    }

    // Read contents from file
    c = fgetc(fptr);
    while (c != EOF)
    {
        printf ("%c", c);
        c = fgetc(fptr);
    }
    fclose(fptr);
    return 0;
}
```

Output:

Enter the filename to open

a.txt

/*Contents of a.txt*/

2. Write a C program to copy one file to another file & while doing so replace all lower case character to their equivalent upper case character.

```
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
int main()
{
    FILE *fp1, *fp2;
    char ch;
    fp1 = fopen("source.txt", "r");
    if (fp1 == NULL)
    {
        puts("File does not exist..");
        exit(1);
    }
    fp2 = fopen("target.txt", "w");
    if (fp2 == NULL)
    {
        puts("File does not exist..");
        fclose(fp1);
        exit(1);
    }
    while((ch=fgetc(fp1))!=EOF)
    {
        ch = toupper(ch);
        fputc(ch,fp2);
    }
    printf("\nFile successfully copied..");
    return 0;
}
```

Output:

Content in source.txt file.

source.txt ✕

```
Welcome to TutorialRide..
```

```
Happy c-programming
```

```
Created By..
```

```
    Sapna
```

All the lower-case letters in source.txt file.

*target.txt ✕

```
WELCOME TO TUTORIALRIDE..
```

```
HAPPY C-PROGRAMMING
```

```
CREATED BY..
```

```
    SAPNA
```

Write a C program to count the number of times a character occurs in a text file. The file name and the character are supplied as command line arguments.

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

void find_frequency(char [], int []);
int main()

{
    char string[100];
    int c, count[26] = {0};
    printf("Input a string\n");
    gets(string);
    find_frequency(string, count);

    printf("Character Count\n");
    for (c = 0 ; c < 26 ; c++)
        printf("%c \t %d\n", c + 'a', count[c]);
    return 0;
}

void find_frequency(char s[], int count[]) {
    int c = 0;
    while (s[c] != '\0')
        {
        if (s[c] >= 'a' && s[c] <= 'z' )
            count[s[c]-'a']++;
            c++;
        }
    }
```

OUTPUT:

Input a String:

ABCDEF

A 1

B 1

C 1

D 1

E 1

F 1

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)

```
#include<stdio.h>
void main()
{
FILE *f1,*f2,*f3; char s[100];
f1=fopen("D:\file3.txt","r");
f2=fopen("D:\file4.txt","r");
f3=fopen("D:\file5.txt","w"); if(f1==NULL || f2==NULL || f3==NULL)
{
printf("error opening file"); exit(0);
}
while(fgets(s,99,f1)!=NULL)
fputs(s,f3);
while(fgets(s,99,f2)!=NULL)
fputs(s,f3);
fclose(f1);
fclose(f2);
fclose(f3);
}
```

OUTPUT:

Before

After

file3.txt file4.txt

file3.txt

file4.txt

file5.txt

ABC
DEF

PQRqrs
t
STUuv

ABC
DEF

PQR
STU

ABC
DEF
PQR
STU

STRINGS

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

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>
void main()
{
    char rom[30];
    int a[30], l, i, k, dec;
    clrscr();
    printf("Enter the roman number\n");
    scanf("%s", &rom);
    l = strlen(rom);
    for(i = 0; i < l; i++)
    {
        switch (rom[i])
        {
            case 'I': a[i] = 1;
                break;
            case 'V': a[i] = 5;
                break;
            case 'X': a[i] = 10;
                break;
            case 'L': a[i] = 50;
                break;
            case 'C': a[i] = 100;
                break;
            case 'D': dec = dec + 500;
                break;
            case 'M': a[i] = 1000;
                break;
            default : printf("Invalid choice");
                break;
        }
    }
    k = a[l - 1];
    for(i = l - 1; i > 0; i--)
    {
        if(a[i] > a[i - 1])
```



```
{
k = k - a[i - 1];
}
if(a[i] <= a[i - 1])
{
k = k + a[i - 1];
}
}
printf("decimal equivalent is %d", k);
getch();
}
```

Input & Output:

Enter the roman number

XIV

Decimal equivalent is 14

2. Write a C program that converts a number ranging from 1 to 50 to Roman equivalent

```
#include <stdio.h>
void predigit(char num1, char num2);
void postdigit(char c, int n);
char romanval[1000];
int i = 0;
int main()
{
    int j;
    long number;

    printf("Enter the number: ");
    scanf("%d", &number);
    if (number <= 0)
    {
        printf("Invalid number");
        return 0;
    }
    while (number != 0)
    {
        if (number >= 1000)
        {
            postdigit('M', number / 1000);
            number = number - (number / 1000) * 1000;
        }
        else if (number >= 500)
        {
            if (number < (500 + 4 * 100))
            {
                postdigit('D', number / 500);
                number = number - (number / 500) * 500;
            }
            else
            {
                predigit('C', 'M');
                number = number - (1000-100);
            }
        }
        else if (number >= 100)
```

```

{
  if (number < (100 + 3 * 100))
  {
    postdigit('C', number / 100);
    number = number - (number / 100) * 100;
  }
  else
  {
    predigit('L', 'D');
    number = number - (500 - 100);
  }
}
else if (number >= 50 )
{
  if (number < (50 + 4 * 10))
  {
    postdigit('L', number / 50);
    number = number - (number / 50) * 50;
  }
  else
  {
    predigit('X','C');
    number = number - (100-10);
  }
}
else if (number >= 10)
{
  if (number < (10 + 3 * 10))
  {
    postdigit('X', number / 10);
    number = number - (number / 10) * 10;
  }
  else
  {
    predigit('X','L');
    number = number - (50 - 10);
  }
}
else if (number >= 5)
{

```

```

    if (number < (5 + 4 * 1))
    {
        postdigit('V', number / 5);
        number = number - (number / 5) * 5;
    }
    else
    {
        predigit('I', 'X');
        number = number - (10 - 1);
    }
}
else if (number >= 1)
{
    if (number < 4)
    {
        postdigit('I', number / 1);
        number = number - (number / 1) * 1;
    }
    else
    {
        predigit('I', 'V');
        number = number - (5 - 1);
    }
}
}
printf("Roman number is: ");
for(j = 0; j < i; j++)
    printf("%c", romanval[j]);
return 0;
}

```

```

void predigit(char num1, char num2)
{
    romanval[i++] = num1;
    romanval[i++] = num2;
}

```

```

void postdigit(char c, int n)
{
    int j;

```

```
for (j = 0; j < n; j++)  
    romanval[i++] = c;  
}
```

Output:

Enter the number: 500

Roman number is be: D

- i. Write a C program that uses functions to perform the following operations:

To insert a sub-string in to a given main string from a given position.

Program:

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

void insertStr(char m[100],char s[100],int pos);
int main()
{
char m[100],s[100]; int pos,n;
printf("\n enter main string for insertion:");
gets(m);
printf("\n enter sub string:");
gets(s);
printf("\nEnter position:");
scanf("%d",&pos);
insertStr(m,s,pos);
printf("\nMain string after insertion: %s",m);
getch();
return(0);
}
void insertStr(char m[100],char s[100],int pos)
{
int i,lm=strlen(m),ls=strlen(s);
for(i=lm;i>=pos;i--)
m[i+ls]=m[i];
for(i=0;i<ls;i++)
m[i+pos]=s[i];
}
```

OUTPUT:

```
enter main string for insertion:comer
enter sub string:put
Enter position:3
Main string after insertion: computer
```

To delete n Characters from a given position in a given string.

```
#include<stdio.h>
#include<string.h>
void deleteStr(char m[100],int n,int pos);
int main()
{
    char m[100],s[100]; int pos,n;
    printf("\nEnter main string for deletion:");
    gets(m);
    printf("\nEnter no of characters to be deleted:");
    scanf("%d",&n);
    printf("\nEnter position:");
    scanf("%d",&pos);
    deleteStr(m,n,pos);
    printf("\nmain string after deletion: %s",m);
    getch();
    return(0);
}

void deleteStr(char m[100],int n,int pos)
{
    int i,Len=strlen(m); for(i=pos;i<=(Len+1-n );i++)
        m[i]=m[i+n];
}
```

OUTPUT:

```
Enter main string for deletion: abcdef
Enter no of characters to be deleted:2
Enter position:3
main string after deletion: abcf
```

Write a C program to determine if the given string is a palindrome or not (Spelled same in both directions with or without a meaning like madam, civic, noon, abcba, etc.)

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

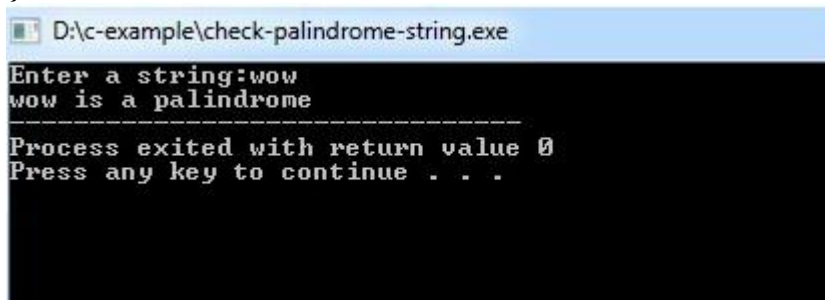
int main(){
    char string1[20];
    int i, length;
    int flag = 0;

    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;
}
```



```
D:\c-example\check-palindrome-string.exe
Enter a string:wow
wow is a palindrome
-----
Process exited with return value 0
Press any key to continue . . .
```


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.

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
void main()
{
    char s[30], t[20];
    char *found;
    clrscr();
    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);
    }
    else
    {
        printf("-1");
    }
    getch();
}
```

Input & Output:

1. Enter the first string:

kali

Enter the string to be searched:

li

second string is found in the first string at 2 position

2. Enter the first string:

nagaraju

Enter the string to be searched:

raju

second string is found in the first string at 4 position

3. Enter the first string:

nagarjuna

Enter the string to be searched:

Ma -1

Write a C program to count the lines, words and characters in a given text.

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
    char str[100];
    int i = 0, l = 0, f = 1;
    clrscr();
    puts("Enter any string\n");
    gets(str);
    for(i = 0; str[i] != '\0'; i++)
    {
        l = l + 1;
    }
    printf("The number of characters in the string are %d\n", l);
    for(i = 0; i <= l-1; i++)
    {
        if(str[i] == ' ')
        {
            f = f + 1;
        }
    }
    printf("The number of words in the string are %d", f);
    getch();
}
```

Input & Output:

Enter any **string**

abc def ghi jkl mno pqr stu vwx yz

The number of characters **in** the **string** are 34

The number of words **in** the **string** are 9

Miscellaneous:

a.) Write a menu driven C program that allows a user to enter n numbers and then choose between finding the smallest, largest, sum, or average. The menu and all the choices are to be functions. Use a switch statement to determine what action to take. Display an error message if an invalid choice is entered.

PROGRAM:

b.) Write a C program to construct a pyramid of numbers as follows:

```
1      *      1      1      *
1 2    **     2 3    2 2    **
1 2 3  ***    4 5 6    3 3 3  ***
                        |4 4 4  **
                        *

```

Program:

```
#include <stdio.h>

int main()
{
    int i, j, rows;
    printf("Enter number of rows: ");
    scanf("%d",&rows);
    for(i=1; i<=rows; ++i)
    {
        for(j=1; j<=i; ++j)
        {
            printf("* ");
        }
        printf("\n");
    }
    return 0;
}
```

OUTPUT:

Enter number of rows: 5

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

PROGRAM:

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

OUTPUT:

Enter number of rows: 5

```
1  
1 2  
1 2 3  
1 2 3 4  
1 2 3 4 5
```

PROGRAM:

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

OUTPUT:

```
Enter number of rows: 4
1
2 3
4 5 6
7 8 9 10
```

PROGRAM:

```
#include <stdio.h>
int main()
{
    int rows, i, j, number= 1;
    printf("Enter number of rows: ");
    scanf("%d",&rows);
```

```

for(i=1; i <= rows; i++)
{
    for(j=1; j <= i; ++j)
    {
        printf("%d ", number);
    }
    ++number;
    printf("\n");
}
return 0;
}

```

OUTPUT:

Enter number of rows: 4

```

1
2 2
3 3 3
4 4 4 4

```

PROGRAM:

```

#include <stdio.h>

int main()
{
    int i, j, rows;
    printf("Enter number of rows: ");
    scanf("%d",&rows);
    for(i=1; i<=rows; ++i)
    {
        for(j=1; j<=i; ++j)
        {
            printf("* ");
        }
        printf("\n");
    }
}

```

```

}
for(i=rows-1; i>=1; --i)
{
    for(j=1; j<=i; ++j)
    {
        printf("* ");
    }
    printf("\n");
}

```

OUTPUT:

Enter number of rows: 4

```

*
**
***
****
***
**
*

```

Sorting and Searching:

- a.) Write a C program that uses non recursive function to search for a Key value in a given
- b.) List of integers using linear search method.

PROGRAM:

```

#include<stdio.h>
#include<conio.h>
void main()
{
    int i, a[20], n, key, flag = 0;
    clrscr();
    printf("Enter the size of an array \n");
    scanf("%d", &n);
    printf("Enter the array elements");
    for(i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
}

```

```

}
printf("Enter the key elements");
scanf("%d", &key);
for(i = 0; i < n; i++)
{
    if(a[i] == key)
    {
        flag = 1;
        break;
    }
}
if(flag == 1)
    printf("The key elements is found at location %d", i + 1);
else
    printf("The key element is not found in the array");
getch();
}

```

Input & Output:

Enter the size of an array 6
Enter the array elements 50 10 5 200 20 1
Enter the key element 1
The key Element is found at location 6

c.) Sorted list of integers using binary search method.

```

#include <stdio.h>

int BinarySearching(int arr[], int max, int element)
{
    int low = 0, high = max - 1, middle;
    while(low <= high)
    {
        middle = (low + high) / 2;
        if(element > arr[middle])
            low = middle + 1;
        else if(element < arr[middle])
            high = middle - 1;
        else
            return middle;
    }
}

```



```

    }
    return -1;
}
int main()
{
    int count, element, limit, arr[50], position;
    printf("Enter the Limit of Elements in Array:\t");
    scanf("%d", &limit);
    printf("Enter %d Elements in Array: \n", limit);
    for(count = 0; count < limit; count++)
    {
        scanf("%d", &arr[count]);
    }
    printf("Enter Element To Search:\t");
    scanf("%d", &element);
    position = BinarySearching(arr, limit, element);
    if(position == -1)
    {
        printf("Element %d Not Found\n", element);
    }
    else
    {
        printf("Element %d Found at Position %d\n", element, position + 1);
    }
    return 0;
}

```

OUTPUT:

```

Enter the Limit of Elements in Array: 5
Enter 5 Elements in Array:
57

```

```
93
25
57
76
Enter Element to Search:      76
Element 76 Found at Position 5
```

d.) Write a C program that implements the Bubble sort method to sort a given list of integers in ascending order.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int n, a[20], temp, i, j;
    clrscr();
    printf("Enter the size of the array\n");
    scanf("%d", &n);
    printf("Enter the array elements\n");
    for(i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
    for(i = 0; i < n - 1; i++)
    {
        for(j = 0; j < n - 1; j++)
        {
            if(a[j] > a[j + 1])
            {
                temp = a[j];
                a[j] = a[j + 1];
                a[j + 1] = temp;
            }
        }
    }
    printf("The sorted array is\n");
    for(i = 0; i < n; i++)
        printf("%d\n", a[i]);
    getch();
}
```

Output:

Enter the size of the array: 5

Enter the array elements: 50 40 30 20 10

The sorted array is: 10 20 30 40 50

e.) Write a C program that implements the **selection sort** method to sort a given list of integers in descending order.

PROGRAM:

```

#include<stdio.h>
#include<conio.h>
void main()
{
    int n, a[20], min, temp, i, j;
    clrscr();
    printf("Enter the size of the array\n");
    scanf("%d", &n);
    printf("Enter the array elements\n");
    for(i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
    for(i = 0; i < n - 1; i++)
    {
        min = i;
        for(j = i + 1; j < n; j++)
        {
            if(a[j] > a[min])
                min = j;
        }
        temp = a[i];
        a[i] = a[min];
        a[min] = temp;
    }
    printf("The sorted array is\n");
    for(i = 0; i < n; i++)
        printf("%d\n", a[i]);
    getch();
}

```

Output:

Enter the size of the array: 7
Enter the array elements: 1 2 3 4 5 6 7

The Sorted array is: 7 6 5 4 3 2 1

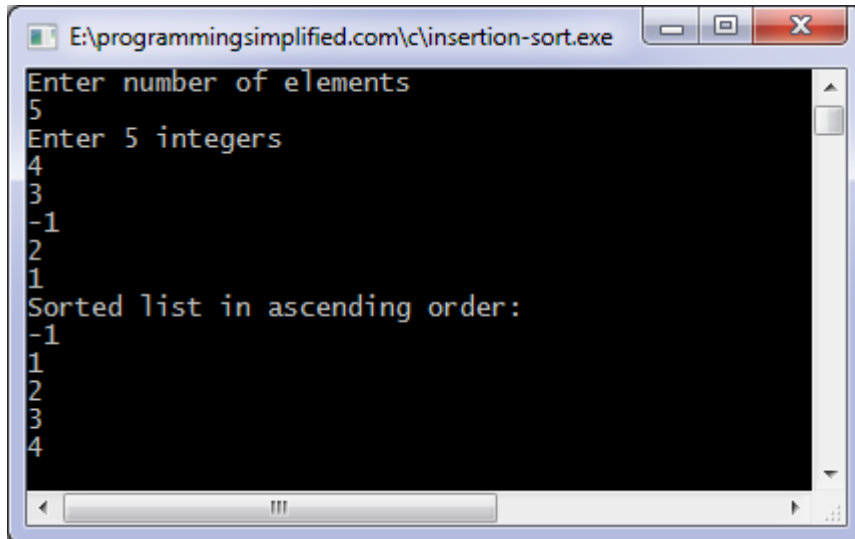
f.) Write a C program that sorts the given array of integers using insertion sort in ascending order .

PROGRAM:

```
# include <stdio.h>
int main()
{
    int n, array[1000], c, d, t;
    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]);
    }
    for (c = 1 ; c <= n - 1; c++)
    {
        d = c;
        while ( d > 0 && array[d-1] > array[d])
        {
            t = array[d];
            array[d] = array[d-1];
            array[d-1] = t;
            d--;
        }
    }
    printf("Sorted list in ascending order:\n");
    for (c = 0; c <= n - 1; c++)
    {
        printf("%d\n", array[c]);
    }
}
```

```
return 0;
}
```

OUTPUT:



```
E:\programmingsimplified.com\c\insertion-sort.exe
Enter number of elements
5
Enter 5 integers
4
3
-1
2
1
Sorted list in ascending order:
-1
1
2
3
4
```

- g)** Write a C program that sorts a given array of names
/** C program to read N names, store them in the form of an array
* and sort them in alphabetical order. Output the given names and
* the sorted names in two columns side by side. */

PROGRAM:

```
#include <stdio.h>
#include <string.h>
void main()
{
    char name[10][8], tname[10][8], temp[8];
    int i, j, n;
    printf("Enter the value of n \n");
    scanf("%d", &n);
```

```

printf("Enter %d names n\n", n);
for (i = 0; i < n; i++)
{
    scanf("%s", name[i]);
    strcpy(tname[i], name[i]);
}
for (i = 0; i < n - 1 ; i++)
{
    for (j = i + 1; j < n; j++)
    {
        if (strcmp(name[i], name[j]) > 0)
        {
            strcpy(temp, name[i]);
            strcpy(name[i], name[j]);
            strcpy(name[j], temp);
        }
    }
}
printf("\n-----\n");
printf("Input NamesSorted names\n");
printf("-----\n");
for (i = 0; i < n; i++)
{
    printf("%s\t\t%s\n", tname[i], name[i]);
}
printf("-----\n");
}

```

OUTPUT:

```
Enter the value of n
```

```
4
```

```
Enter 4 names n
```

```
apple
```

```
boy
```

```
zeebra
```

```
cat
```

```
-----
```

```
Input NamestSorted names
```

```
-----
```

```
apple          apple
```

```
boy            boy
```

```
zeebra        cat
```

```
cat            zeebra
```

```
-----
```