COMPUTER PROGRAMMING

LABORATORY MANUAL AND WORK BOOK

(USING C PROGRAMMING)

2014 - 2015

WEEK- 1

1.1 OBJECTIVE:

1. To find the sum of individual digits of a given number N.
2. Generate the Fibonacci series up to N.
3. To Generate the Prime numbers up to N .

1.2 RESOURCES:

centOS

1.3 PROGRAM LOGIC:

<u>Sum of Individual digits of a given number</u>

1. Read a number N
2. Initialize the Sum=0
3. if n > 0 then calculate reminder R= N %10
4. Add reminder R to Sum
5. Store N/10 value to N
6. If N=0 then Display Sum

<u>Fibonacci Series</u>

1. Initialize first two number of Fibonacci series A=0, B=1
2. Read N value for displaying the Fibonacci series up to Nth term
3. If N= 1 or N=2 display the value of A and B
4. If N > 2, calculate the next term by taking a temporary variable T = A + B
5. Display the term T
6. Swap the contents of A and B using a third variable T

<u>Generate Prime Number</u>

1. Read the prime number range N
2. Initialize flag =0
3. Take two numbers i and j, if i % j =0 then make flag=1
4. If flag = 0 then display the number i as prime number

1.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out.

1.5 SOURCE CODE:

<u>Sum of Individual digits of a given number</u>

```
#include<stdio.h>
void main()
{
        int num, k=1, sum=0;
        printf(“Enter the number whose digits are to be added:”);
        scanf(“%d”, & num);
        while(num!=0)
```

```
    {
         k=num%10;
        sum=sum+k;
        k=num/10;
         num=k;
    }
    printf(“Sum of the digits:%d”,sum);
    getch();
}
```

Fibonacci Series

```
#include<stdio.h>
void main()
{
    int a,b,c,n,i;
    printf("enter n value");
    scanf("%d",&n);
    a=0;
    b=1;
    if(n==1)
        printf("%d",a);
    elseif(n==2)
        printf("%d%d",a,b);
    else
    {
        printf("%d%d",a,b);
        for(i=2;i<n;i++)
        {
            c=a+b;
            printf("%d",c);
            a=b;
            b=c;
        }
    }
}
```

Generate Prime Number

```
#include<stdio.h>
void main()
{
    int n,i,fact,j;
    printf("Enter any number:");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
```

```
                {
                        fact=0;
                        for(j=1;j<=i;j++)
                        {
                                if(i%j==0)
                                fact++;
                        }
                if(fact==2)
                        printf("\n %d",i);
                }
        }
```

1.6 INPUT AND OUTPUT:

Sum of Individual digits of a given number

```
Enter the value for n: 333
Sum of individual digits is 9

Enter the value for n: 4733
Sum of individual digits is 17

 Enter the value for n: -111
The given number is not valid
```

Fibonacci Series

```
Enter n value: 5
0   1   1   2   3
2. Enter n value: 7
0   1   1   2   3   5   8
3. Enter n value: -6
0   1
```

Generate Prime Number

```
Enter the number: 5
2   3    5
Enter the number: 10
2   3   5   7
Enter the number: 12
2   3   5   7
```

1.7 PRE LAB VIVA QUESTIONS:

1. What are the two types of flow control used ?
2. What do you mean by looping ?
3. What is the difference between while and do while loop ?
4. Which loop is known as entry controlled loop ?
5. Which loop is known as exit control loop ?

6. What are the two commands to control the loop ?
7. In which loop the test condition is tested at the end of the loop ?

1.8 LAB ASSIGNMENT:

1. Find the sum of the individual digits of a number ?
2. Check whether the given number is an Armstrong number or not ?
3. Display the reverse of a given number.
4. Display the series of numbers 1, 3, 5, 7, 9………………..
5. Display all prime numbers between 1 to N.
6. Generate all combinations of 1, 2, 3 using for loop ?
7. Display the following series using for loop

   1 + 2/2! + 3/3! +4/4! ………………………………………..
8. Display all the ASCII values and their equivalent characters using while loop
9. Display the perfect numbers between 1 to N.
10. Display the friendly numbers between 1 to N.

1.9 POST LAB VIVA QUESTIONS:

1. Can we initialize more than one variable in for loop ?
2. Give one example of for loop without increment/decrement operation ?
3. Is the following code for(;;) represents an infinite loop ?
4. What is the order of movement of control in for loop ?

```
for( expression1; expression2; expression3)
{
     Block of statements;
}
```

5. Can we write more than one test condition in for loop ?
6. What is the output of the following code ?

```
#include<stdio.h>
void main()
{
            int i=1;
             for(;i++;)
            printf("%d",i);
}
```

WEEK - 2

2.1 OBJECTIVE:

1. To calculate the Sum=1-x^2/2! +x^4/4!-x^6/6! +x^8/8!-x^{10}/10!
2. To find the roots of a quadratic equation.

2.2 RESOURCES:

centOS

2.3 PROGRAM LOGIC:

Sum=1-x^2/2! +x^4/4!-x^6/6! +x^8/8!-x^{10}/10!

1. Read x value.
2. calculate sum=1-x^2/2! +x^4/4!-x^6/6! +x^8/8!-x^{10}/10!.
3. print sum.

Roots of Quadratic Equation

1. Read the coefficients of a quadratic equation a, b, c
2. Calculate determinant d = b*b – 4*a*c
3. If d > 0 calculate two real roots r1 = (-b + sqrt(d)) / (2*a) and r2 = (-b + sqrt(d)) / (2*a)
4. If d=0 then roots r1 and r2 are equal and display r1 = r2 = -b / (2*a)
5. If d < 0 then roots are imaginary and display real root= -b /(2 * a) and img root =sqrt(-d) / (2*a)

2.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile : gcc x.c.
3. Execute : . / a.out.

1.5 SOURCE CODE:

Sum=1-x^2/2! +x^4/4!-x^6/6! +x^8/8!-x^{10}/10!

```
#include<stdio.h>
#include<math.h>
void main()
{
    int i = 2, n, s = 1, x, pwr = 1, dr;
    float nr = 1, x1, sum;
    clrscr();
    printf("\n\n\t enter the angle...: ");
    scanf("%d", &x);
    x1 = 3.142 * (x / 180.0);
    sum = x1;
    printf("\n\t enter the number of terms...: ");
    scanf("%d", &n);

            while ( i <= n )
            {
               pwr = pwr + 2;
               dr = dr * pwr * (pwr - 1);
               sum = sum + (nr / dr) * s;
            s = s * (-1);
            nr = nr * x1 * x1;
            i+= 2;
        }
        printf("\n\t the sum of the sine series is..: %0.3f",sum);
        getch();

}
```

Roots of Quadratic Equation

```
#include<stdio.h>
#include<math.h>
void main()
{
        float a,b,c,r1,r2,d;
        printf("Enter the values for equation:");
        scanf("%f%f%f",&a,&b,&c);
        if(a==0)
                printf("Enter value should not be zero ");
        else
        {
                d=b*b-4*a*c;
                if(d>0)
                {
                        r1=(-b+sqrt(d)/(2*a));
                        r2=(-b-sqrt(d)/(2*a));
                        printf("roots are real and unequal\n");
                        printf("%f\n%f\n",r1,r2);
                }
                elseif(d==0)
                {
                        r1=-b/(2*a);
                        r2=-b/(2*a);
                        printf("roots are real and equal\n");
                        printf("root=%f\n",r1);
                        printf("root=%f\n",r2);
                }
                else
                        printf("roots are imaginary");
        }
}
```

2.6 INPUT/OUTPUT:

Sum=$1-x^2/2! +x^4/4!-x^6/6! +x^8/8!-x^{10}/10!$

```
Enter the angle...: 30
 Enter the number of terms: 3
 The sum of the sine series is..:0.524
```

Roots of Quadratic Equation

```
Enter the values for equation: 1, 6, 9
Roots are real and equal
Root= -3.0000
Root= -3.0000
Enter the values for equation: 2, 7, 6
Roots are real and unequal
Root= -6.75
Root= -7.25
Enter the values for equation: 1, 2, 3
Roots are imaginary
```

2.7 PRE LAB VIVA QUESTIONS:

1. What is a prime number ?
2. Which header file is required to use mathematical function like sqrt() ?
3. Which operator is used to find the factors of a number ?
4. Which operator is used to find the negation of the number ?
5. Which decision making statement is used to check the determinant value of a quadratic equation ?
6. Using which operator we can write multiple conditions in an if statement ?
7. What type value is returned after evaluating the condition(s) in an if statement ?

2.8 LAB ASSIGNMENT:

1. Any character is entered through the keyboard, write a program to determine whether the character entered is a capital letter, a small case letter, a digit, or a special symbol.
2. Display the following in triangular form

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

3. Write a program to display the triangular number where a triangular number is nothing but summation of 1 to given no N.

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

4. Read a 3 digit number N from keyboard and find individual digits in unit's place (U), ten's place (T) and hundred's place (H). check $U^3 + T^2 + H = N$ (given no)
5. Find the frequency of a particular character in a string ?
6. Check whether a year is leap year or not ?

7. Display the largest among three numbers using if else statement ?
8. Calculate the sum of natural numbers 1+2+3+4+……………..+n
9. Check whether the entered character is vowel or consonant ?
10. Find the number of digits of a given integer ?

2.9 POST LAB VIVA QUESTIONS:

1. Can you write for loop without body, if yes then what is the output of the following code ?

```
#include<stdio.h>
int main()
{
   int i;
    for(i=0;i<=10;i++);
   printf("%d",i);
 return 0;
}
```

2. What is the output of the following program ?

```
#include<stdio.h>
int main()
{
        int i,j=2;
        for(i=0;j>=0,i<=5;i++)
       {
                    printf("%d ",i+j);
                    j--;
        }
return 0;
}
```

3. What is the output of the following program ?

```
#include<stdio.h>
int main()
{
    int i;
    for(;i=0,i<=3 ;i++){
         printf("%d ",i);
    }
return 0;
}
```

4. What is the output of the following program ?

```
#include<stdio.h>
int main()
{
    int i;
    for(i=0;-5 ;i++)
    {
        printf("%d ",i);
        if(i==3)
            break;
    }

return 0;
}
```

5. What is the output of the following code ?

```
#include <stdio.h>
 int main()
 {
        unsigned int a = 10;
        a = ~a;
        printf("%d\n", a);
 }
```

6. What is the output of the following code ?

```
#include<stdio.h>
int i=40;
extern int i;
int main()
{
    do{
            printf("%d",i++);
    } while(5,4,3,2,1,0);
    return 0;
}
```

7. What is the output of the following code ?

```
#include<stdio.h>
int main(){
    int i;
    for(i=10;i<=15;i++){
            while(i){
```

```
                                        do{
                                                printf("%d ",1);
                                                if(i>>1)
                                                                continue;
                                        }while(0);
                                        break;
                                }
                }
                return 0;
        }
```

WEEK- 3

3.1 OBJECTIVE:

1. The total distance traveled by vehicle in ‘t’ seconds is given by distance=ut+1/2at2 where ‘u’ and ‘a’ are the initial velocity (m/sec) and acceleration (m/sec2). Write a C program to find the distance traveled at regular intervals of time given values of ‘u’ and ‘a’. The program should provide the flexibility to the user to select his own time intervals and repeat the calculations for different values of ‘u’ and ‘a’.

2. Using switch-case statement, write a C program that takes two operands and one operator from the user, performs the operation and then prints the answer. (consider operators +, -, *, and %).

3.2 RESOURCES:

centOS

3.3 PROGRAM LOGIC:

Distance calculation:

1. Read initial velocity u, acceleration a, and time t.
2. Calculate the distance travelled by the vehicle S= u * t + (1 / 2 * (a * t * t))
3. Display the distance travelled S.

Arithmetic Operations using switch case

1. Read two numbers a and b
2. Read your choice of operator ch
3. If ch = ‘+’ then calculate add = a + b and display the addition result.
4. If ch= ‘-‘ then calculate sub = a – b and display the subtraction result.
5. If ch= ‘*’ then calculate mul = a * b and display the multiplication result.
6. If ch= ‘/’ then calculate div = a / b and display the division result.
7. If ch=’%’ then calculate mod = a % b and display the modulus result.
8. Otherwise display invalid operator

3.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out.

3.5 SOURCE CODE:

Distance calculation

```
include<stdio.h>
void main()
```

```
{
        int a,u,t,t1,t2,i;
        float s;
        printf("ENTER THE VALUES OF a,u,t,t1,t2:");
        scanf("%d%d%d%d%d",&a,&u,&t,&t1,&t2);
        for(i=t1;i<=t2;i=i+t)
        {
                s=(u*i)+(0.5*a*i*i);
                printf("\n\nthe distance travelled in %d seconds is %f ",i,s);
        }
}
```

Arithmetic Operations using switch case

```
#include<stdio.h>
void main()
{
        char op;
        float a,b,c;
        printf("enter two operands:");
        scanf("%d%d",&a,&b);
        printf("enter an operator:");
        scanf(" %c",&op);
        switch(op)
        {
                case '+':printf("sum of two numbers %2d %2d is: %d",a,b,a+b);
                break;
                case '-':printf("subtraction of two numbers %2d %2d is:%d",a,b,a-b);
                break;
                case '*':printf("product of two numbers %2d %2d is: %d",a,b,a*b);
                break;
                case '/':printf("quotient of two numbers %2d %2d is:%d",a,b,a/b);
                break;
                case '%':printf("reminder of two numbers %2d %2d is:%d",a,b,c);
                break;
                default:printf("please enter correct operator");
        }
}
```

3.6 INPUT/OUTPUT:

Distance calculation

```
ENTER THE VALUES OF a,u,t,t1,t2:  1   2   3   1   5
The distance travelled in 1 seconds is 2.500000
The distance travelled in 4 seconds is 16.000000
ENTER THE VALUES OF a,u,t,t1,t2:  0   1   2   3   4
```

```
The distance travelled in 3 seconds is 3.000000
```

Arithmetic Operations using switch case

```
Enter two operands:2 3
enter an operator:+
sum of two numbers 2 3 is: 5

enter two operands:3 4
enter an operator: -
subtraction of two numbers 3 4 is: -1

enter two operands:3 5
enter an operator:*
product of two numbers 3 5 is: 15

enter two operands:5 2
enter an operator:/
quotient of two numbers 5 2 is: 2

enter two operands:5 2
enter an operator:%
reminder of two numbers 5 2 is: 1
```

3.7 PRE-LAB VIVA QUESTIONS:

1. What is an identifier ?
2. What type of information a storage class gives ? What are the different storage classes in C ?
3. What are the types of constants in C ?
4. Which operator is having highest priority and which operator is having least priority in C ?
5. What is the associativity of assignment operators ?
6. Which operator is used to find the size of a data type ?
7. What do you mean by explicit type casting ? Give one example.

3.7 LAB ASSIGNMENT:

1. Compute the area and circumference of a circle ?
2. Check whether a number is even or odd using ternary operator ?
3. Check the greater of two numbers using conditional operator ?
4. Swap the given two numbers without using the third variable ?
5. Check the given number is a palindrome or not ?
6. Display the right most digit of any integer ?
7. Write a program that calculates the sale given the unit price, quantity, discount rate and sales tax ?
8. Write a program that asks the user to enter a temperature reading in Fahrenheit and then prints the equivalent Celsius value by using the following formula

   Celsius = (100 / 180) * (Fahrenheit – 32)

9. Write a C program to print all the even and odd numbers from 1 to 10 and the sum of all even numbers and odd numbers individually ?
10. Write a program to calculate the simple interest S = PTR where P is the principal amount, T is the time period and r is the rate of interest.
11. For the following given string 'COMPUTER PROGRAMMING', formulate a C-program for converting given string into lowercase characters and print them.
12. Formulate a program for converting a decimal representation of a number to the corresponding character string representation.
13. Devise a program that converts a decimal integer into its equivalent octal representation.
14. Given a character representation of an integer, devise a C-program to convert it to its conventional decimal format. (If you give a 'A' as character, its equivalent decimal value 65 should be printed).
15. Formulate a program that reads in a set of n single digits and converts them into a single decimal integer. For example, the algorithm should convert the set of 5 digits {2, 7, 4, 9, 3} to the integer 27493.

3.8 POST-LAB VIVA QUESTIONS:

1. Do you think associativity has no role to play unless the precedence of operators is same ?
2. Why the right hand side of || operators doesn't get evaluated if the left hand side determines the outcome ?
3. Left shift and right shift are the two substitutes for which arithmetic operators in C ?
4. What types of values are returned by relational operators in C ?
5. How many bitwise operators are there in C ?
6. In C which operator = and == has got highest priority ?

Week- 4

4.1 OBJECTIVE:

Program to check

1. the factorial of a given number N
2. the GCD of two numbers

4.2 RESOURCES:

centOS

4.3 PROGRAM LOGIC:

Factorial of a given number by using Recursive function

1. Read a number N
2. Call a function factorial(N) by passing the values of N
3. If N=1 then it returns 1 as the factorial
4. Otherwise it calculates the factorial f = N * factorial(N-1) by calling the same function again and again
5. Display the factorial of number N

Factorial of a given number by using Non-Recursive function

1. Read a number N
2. Initialize fact =1
3. Calculate fact = fact * N using a loop and decrement N value till N =1
4. Display the factorial of number N

GCD of a given two integer by using Recursive Function

1. Read two numbers a and b
2. Call a function gcd (a, b) by passing the value of a and b
3. Check a != b, then
4. Check a > b then calculate gcd of two numbers by calling the function gcd(a-b, b)
5. Otherwise calculate gcd of two numbers by calling the function gcd(a, b-a)
6. Display the gcd of two numbers

GCD of a given two integer by using Non-Recursive Function

1. Read two numbers a and b
2. Check if a != b then
3. Check if a >= b - 1 then set a = a – b otherwise set b = b – a
4. Display the gcd as the value of a.

4.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.

2. Compile: gcc x.c.

3. Execute: . / a.out.

4.5 SOURCE CODE:

Factorial of a given number by using Recursive function

```
#include<stdio.h>
int fact(int n)
{
        int f;
        if((n==0)||(n==1))
                return(n);
        else
                f=n*fact(n-1);
        return(f);
}
void main()
{
        int n;
        printf("enter the number :");
        scanf("%d",&n);
        printf("factoria of number%d",fact(n));
}
```

<u>Factorial of a given number by using Non-Recursive function</u>

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

<u>GCD of a given two integer by using Recursive Function</u>

```
#include<stdio.h>
int gcdrecursive(int m,int n)
{
        if(n>m)
                return gcdrecursive(n,m);
        if(n==0)
```

```
                    return m;
          else
                    return gcdrecursive(n,m%n);
}
void main()
{
          int a,b,igcd;
          printf("enter the two numbers whose gcd is to be found:");
          scanf("%d%d",&a,&b);
          printf("GCD of a,b is %d",gcdrecursive(a,b));
}
```

GCD of a given two integer by using Non-Recursive Function

```
#include<stdio.h>
void main()
{
          int n1,n2;
          printf("To calculate gcd of two numbers without recursion");
          printf("\nEnter two numbers:");
          scanf("%d %d",&n1,&n2);
          while(n1!=n2)
          {
                    if(n1>=n2-1)
                              n1=n1-n2;
                    else
                              n2=n2-n1;
          }
          printf("\nGCD=%d",n1);
}
```

4.6 INPUT/OUTPUT:

Factorial of a given number by using Recursive/Non Recursive function

```
Enter the number : 5
Factorial of number: 120

Enter the number : 3
Factorial of number: 6
```

GCD of a given two integer by using Recursive / Non Recursive Function

```
enter the two numbers whose gcd is to be found:5, 25
GCD of a,b is : 5
 enter the two numbers whose gcd is to be found:36, 54
GCD of a,b is : 18
enter the two numbers whose gcd is to be found:11, 13
GCD of a,b is : 1
```

4.7 PRE LAB VIVA QUESTIONS:

1. What is recursion ?
2. What is the necessity of creating a user defined function in C ?
3. What are the types of functions according to return values and number of arguments ?
4. Can you pass a parameter to a function ? If yes, then what are the parameter passing techniques ?
5. Which data structure is used in recursion ?
6. Give any two differences between recursion and iteration ?
7. What do mean by base / stopping criteria in recursion ? What is its importance ?

4.8 LAB ASSIGNMENT

1. Display the Fibonacci series using Recursive function ?
2. Generate the prime numbers between 1 to N using recursion ?
3. Display the reverse of your name using recursion ?
4. Compute the x to the power y using recursion ?
5. Check whether a string is palindrome or not using recursion ?
6. Compute the sum of digits of a number using recursion ?
7. Compute the sum of n natural numbers using recursion ?
8. Convert a decimal number into its equivalent binary using recursion ?
9. Compute the multiplication of two matrices using recursion ?
10. Perform binary search using recursion ?
11. Write a program to convert decimal number to hexadecimal number ?
12. Write a program to convert decimal number to binary number ?

4.9 POST LAB VIVA QUESTIONS:

1. What do you mean by formal and actual parameters in a function ?
2. What is upward and downward communication in functions ?
3. Can you pass an entire structure into a function ?
4. Can you pass an array to a function ?
5. What do you mean by function prototype and why it is used ?
6. Can you write a function without any parameters ?
7. Can you return multiple values from a function using return statement ?
8. Is it mandatory to write a return statement in a function ?
9. What are the various ways of writing return statement in a function ?
10. Is it possible convert all iterative programs into recursion?

WEEK-5

5.1 OBJECTIVE:

A program that uses functions to perform the following

a) largest number from given list of integers.

b) Addition of Two Matrices

c) Multiplication of Two Matrices

5.2 RESOURCES:

centOS

5.3 PROGRAM LOGIC:

Displaying the largest number among a list of integers

1. Read N value as the number of elements you want to store in the array.
2. Read N different integers and store in the array using a loop.
3. Assume max = a[0] and min = a[0] i.e. the first element of the array.
4. Check the max value with the rest of the elements in the array.
5. If any element is bigger than assumed max value then swap both the values.
6. Display the biggest number.
7. Check the min value with the rest of the elements in the array.
8. If any element is smaller than assumed min value then swap both the values.
9. Display the smallest number.

Matrix addition

1. Read the no. of rows (r1) and cols (c1) of a matrix a[3][3].
2. Read the no. of rows (r2) and cols. (c2) of matrix b[3][3].
3. If c1 = c2 and r1 = r2 then read the elements into both the matrices a and b.
4. Add the elements of first matrix with the elements of second matrix according to their index position.
5. Display the resultant matrix.

Matrix multiplication

1. Read the no. of rows (r1) and cols (c1) of a matrix a[3][3].
2. Read the no. of rows (r2) and cols. (c2) of matrix b[3][3].
3. If c1=r2 then display matrix multiplication is possible otherwise display impossible
4. If c1=r2 then read the elements into both the matrices a and b.
5. Initialize a resultant matrix c[3][3] with 0.
6. Calculate c[i][j] = c[i][j] + a[i][k] * b[k][j].
7. Display the resultant matrix.

5.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out.

5.5 SOURCE CODE:

Displaying the largest among a list of integers

```
#include<stdio.h>
void main()
{
        int a[10],i,n,min,max;
        printf("enter the array size:");
        scanf("%d",&n);
        printf("Enter the elements of array");
        for(i=0;i<n;i++)
        scanf("%d",&a[i]);
        min=a[0];
        max=a[0];
        for(i=0;i<n;i++)
        {
                if(a[i]<min)
                        min=a[i];
                if(a[i]>max)
                        max=a[i];
        }
        printf("maximum value is:%d\n",max);
        printf("minimum value is:%d\n",min);
}
```

Matrix addition

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int m1[10][10],m2[10][10],c1,r1,c2,r2,i,j;
        printf("Enter the no of columns of M1:");
        scanf("%d",&c1);
        printf("Enter the no of rows of M1:");
        scanf("%d",&r1);
        printf("Enter the no of columns of M2:");
        scanf("%d",&c2);
        printf("Enter the no of rows of M2:");
        scanf("%d",&r2);
        if(c1==c2&&r1==r2)
        {
                printf("\nEnter the %d elements of m1:",c1*r1);
                for(i=0;i<r1;i++)
                {
                        for(j=0;j<c1;j++)
                                scanf("%d",&m1[i][j]);
                }
                printf("\nEnter the %d elements of m2:",c2*r2);
                for(i=0;i<r2;i++)
```

```
                    {
                            for(j=0;j<c2;j++)
                                    scanf("%d",&m2[i][j]);
                    }
                    printf("The first matrix is:\n");
                    for(i=0;i<r1;i++)
                    {
                            for(j=0;j<c1;j++)
                                    printf("%3d",m1[i][j]);
                            printf("\n\n");
                    }
                    printf("The second matrix is:\n");
                    for(i=0;i<r2;i++)
                    {
                    for(j=0;j<c2;j++)
                    printf("%3d",m2[i][j]);
                    printf("\n\n");
                    }
                    printf("The result matrix is:\n");
                    for(i=0;i<r2;i++)
                    {
                    for(j=0;j<c2;j++)
                    printf("%3d",m1[i][j]+m2[i][j]);
                    printf("\n\n");
                    }
            }
            else
                    printf("Matrices addition is not possible");
    }
```

Matrix multiplication

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int i,k,j,m1[10][10],m2[10][10],p[10][10],c1,r1,c2,r2;
        printf("Enter the no of rows of M1:");
        scanf("%d",&r1);
        printf("Enter the no of columns of M1:");
        scanf("%d",&c1);
        printf("Enter the no of rows of M2:");
        scanf("%d",&r2);
        printf("Enter the no of columns of M2:");
```

```
scanf("%d",&c2);
if(c1==r2)
{
        printf("Matrices multiplication is possible");
        printf("\n\nEnter the elements of M1:");
        for(i=0;i<r1;i++)
        {
                for(j=0;j<c1;j++)
                        scanf("%d",&m1[i][j]);
        }
        printf("\nEnter the elements of M2:");
        for(i=0;i<r2;i++)
        {
                for(j=0;j<c2;j++)
                        scanf("%d",&m2[i][j]);
        }
        printf("The first matric is:\n");
        for(i=0;i<r1;i++)
        {
                for(j=0;j<c1;j++)
                        printf("%4d",m1[i][j]);
                printf("\n");
        }
        printf("The second matric is:\n");
        for(i=0;i<r2;i++)
        {
                for(j=0;j<c2;j++)
                        printf("%4d",m2[i][j]);
                printf("\n");
        }
        printf("\nThe result matric is:\n");
        for(i=0;i<r1;i++)
        {
                for(j=0;j<c2;j++)
                {
                        p[i][j]=0;
                        for(k=0;k<c1;k++)
                        {
                                p[i][j]+=(m1[i][k]*m2[k][j]);
                        }
                }
        }
```

```
                for(i=0;i<r1;i++)
                {
                        for(j=0;j<c2;j++)
                        {
                                printf("%4d ",p[i][j]);
                        }
                        printf("\n");
                }
        }
        else
                printf("\nMatrices multiplication is not possible");
}
```

5.6 INPUT/OUTPUT:

Displaying the largest among a list of integers

```
enter the array size:4
Enter the elements of array 36   13    2    45
maximum value is:45
minimum value is:2

enter the array size:5
Enter the elements of array 6    2    1    3    8
maximum value is:8
minimum value is:1

enter the array size:5
Enter the elements of array   -6     9    -9     2    5
maximum value is:9
minimum value is:-9
```

Matrix addition

```
Enter the no of columns of M1:3
Enter the no of rows of M1:3
Enter the no of columns of M2:3
Enter the no of rows of M2:3
Enter the 9 elements of m1:1 2 3 4 5 6 7 8 9
Enter the 9 elements of m2:1 2 3 4 5 6 7 8 9
The first matric is:
  1    2    3
  4    5    6
  7    8    9
```

```
The second matric is:
 1   2   3
 4   5   6
 7   8   9
The result matric is:
 2    4    6
 8   10   12
14   16   18
Matrix multiplication
Enter the no of rows of M1:2
Enter the no of columns of M1:3
Enter the no of rows of M2:3
Enter the no of columns of M2:2
Matrices multiplication is possible
Enter the elements of M1:1 2 3 4 5 6
Enter the elements of M2:1 2 3 4  5 6
The first matrix is:
        1  2  3
        4  5  6
The second matrix is:
        1  2
        3  4
        5  6
The result matrix is:
 22  28
 49  64
```

5.7 PRE-LAB VIVA QUESTIONS:

1. What is the purpose of the following instructions:
2. When do you use two-dimensional character array ?
3. Differentiate between standard functions & user-defined functions ?
4. How does the function definition differ from function declaration ?
5. When dos the compiler not implicitly generate the address of first element of Array ?

5.8 LAB ASSIGNMENT:

1. Read number of rows and columns for 2 matrices. Perform addition of two matrices and display the result.
2. Formulate a C-programs that makes the following tasks:
   a. Check whether Matrix is Magic Square or Not ?
   b. Print Square of Each Element of 2D Array Matrix
   c. Find Transpose of Given Square Matrix
   d. Find addition of Lower Triangular Elements in C Programming

e. Calculate sum of Upper Triangular Elements in C

f. Evaluate Subtraction of two matrices (matrix) in C

g. Addition of Diagonal Elements in Matrix

h. Addition of All Elements in Matrix

i. Accessing 2-D Array Elements In C Programming

j. Find Inverse Of 3 x 3 Matrix in 10 Lines

k. Multiply Two 3 X 3 Matrices

3. Given a set of n students examination marks (in the range 0 to 100), Formulate a program that makes a count of the number of students that passed the examination. A pass is awarded for all marks of 50 and above.
4. Compose a program that reads a list of n numbers and makes a count of the number of negatives and the number of non-negative members in the set.
5. Formulate a C-program that reads a list of n numbers and makes a count of the number of even numbers, odd numbers.
6. Given an array of n elements. Read n elements into array. Formulate a C-program for finding smallest element in array using pointers.

5.9 POST-LAB VIVA QUESTIONS:

1. In C, if you pass an array as an argument to a function, what actually gets passed ?
2. Do you think a pointer to a block of memory is effectively same as an array ?
3. What is the difference in the following declarations

```
int fun(int a[1]);
int fun(int a[2]);
```

4. What is output of the following code ?

```
#include<stdio.h>
int main()
{
        int arr[]={0,10,20,30,40};
        char *ptr=arr;
        ptr=ptr+2;
        printf("%d",*ptr);
        return 0;
}
```

5. What is condition for performing a matrix multiplication ?
6. What is the output of the following code ?

```
#include<stdio.h>
int main()
{
        int a=5,b=8,c=2;
        int *arr[]={&a,&b,&c};
         b=a+c;
         printf("%d",*arr[1]);
        return 0;
}
```

7. What is the output of the following code ?

```
#include<stdio.h>
int main()
{
        int a[5]={100,200,300};
        int *p1=a;
        char *p2=(char *)a;
        printf("%d   %d",*(p1+2),*(p2+3));
        return 0;
}
```

WEEK-6

6.1 OBJECTIVE:

Interpret a programs using functions

1. To insert a sub string into given main string from a given position
2. To delete n characters from a given position in a given string

6.2 RESOURCES:

centOS

6.3 PROGRAM LOGIC:

<u>To insert a sub string into given main string from a given position</u>

1. Read the main string into an array a[50].
2. Read the sub string into another array b[30].
3. Enter the position at which we want to insert the substring into main string.
4. Initialize another character array c[80].
5. If pos != 0 then copy the characters from array a[50] to array c[80] till it reaches to pos – 1.
6. From the current position copy the contents of the array b[30] till end to third array c[80].
7. Copy the remaining characters from the first array a[50] till the end to the third array c[80].
8. Display the contents of the array c[80].

<u>To delete n characters from a given position in a given string</u>

1. Read a string into a character array a[50].
2. Read the position from where it has to delete the characters pos.
3. Read the no. of characters to be deleted n.
4. Initialize another character array b[50].
5. If pos != 0 then copy the characters from array a[50] to array b[50] till it reaches to pos-1.
6. Skip n characters from the current position by making pos = pos + n.
7. Copy the remaining characters from the new position of array a[50] till the end to b[50];
8. Display the new string from array b.

6.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out.

6.5 SOURCE CODE:

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
        char a[10], b[10], c[10];
        int p=0,r=0,i=0,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;
        while(i <= r)
        {
                c[i]=a[i];
                i++;
        }
        s = n+r;
        o = p+n;
        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);
}
```

To delete n characters from a given position in a given string

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void string(char a[20],int p,int n)
{
        int i=0,j=0;
        char s[20];
        while(i<p)
        {
                s[j]=a[i];
                i++;
                j++;
        }
        i=i+n;
        while(a[i]!=0)
        {
                s[j]=a[i];
                j++;
                i++;
        }
        s[j]=0;
        printf("New string is %s",s);
}
void main()
{
        char a[50];
        int p,n;
        printf("Enter the string : ");
        scanf("%s",&a);
        printf("\nEnter the position from where it has to be deleted : ");
        scanf("%d",&p);
        printf("\nEnter the no.of characters to be deleted : ");
        scanf("%d",&n);
        string(a,p,n);
}
```

6.6 INPUT/OUTPUT:

To insert a sub string into given main string from a given position

```
enter first string:
rama
enter second string:
```

```
krishna
enter the position where the item has to be inserted:4
ramakrishnarao
```

To delete n characters from a given position in a given string

```
Enter the string : allani.saikukumar
Enter the position from where it has to be deleted : 10
Enter the no. of characters to be deleted : 2
New string is allani.saikumar
```

6.7 PRE-LAB VIVA QUESTIONS:

1. How can I convert a string to a number ?
2. Is it possible to execute code even after the program exits the main() function ?
3. Name the different string handling functions ?
4. Which command is used to combine the two strings ?
5. Which command is used to copy the strings ?
6. What are the various types of string functions ?

6.8 LAB ASSIGNMENT:

1. Code a program to convert the given string into lower case to upper case and vice-versa.
2. Read two strings and store them in variables str1, str2. Now formulate a C-program for finding whether the both strings are equal or not.
3. Program for finding the length of the given string “COMPUTER PROGRAMMING” with and without using string length function.
4. Read two strings and store them in variables str1, str2. Now formulate a C-program for copying 2 strings to a new variable str3.
5. Formulate a program for concatenating the following two strings with and without using strcat() function. Str1 is ‘COMPUTER’, str2 is ‘PROGRAMMING’.
6. Formulate a program for reversing the given string ‘REVERSE’ with and without using stirng reverse function.
7. Formulate a program for determining whether the given string is palindrome or not ?

6.9 POST-LAB VIVAQUESTIONS:

1. What is the difference between string copy(strcpy()) and memory copy (memcpy()) ? When should each be used ?
2. How can I copy just a portion of a string ?
3. How can I convert a number to a string ?
4. How can you tell whether two strings are same ?
5. How do you print only part of a string ?

Week-7

7.1 OBJECTIVE:

1. Displays the position or index in the string S where the string T begins
2. To count the lines, words and characters in a given text.

7.2 RESOURCES:

centOS

7.3 PROGRAM LOGIC:

<u>Display the position of the string</u>

1. Read a string into a character array a[50].
2. Read a substring into another array b[50].
3. Check whether the substring is present in the main string or not using strstr() function.
4. If the substring is present then it returns the address of substring in main string and from there we can display the position.
5. Otherwise display -1 means the substring is not present in the main string.

To count the lines, words & characters in a given text

1. Read the text into a character array txt[100].
2. Initialize word=0, ch=0 and lines=0;
3. Place a special character like $ at the end of the text.
4. Read character by character and check if ch = 32 || ch = ’\n’ ||ch = ‘,’ || ch=’.’ then count as words.
5. If ch=’\n’ then count as lines.
6. Find the string length of the text and count as ch.
7. Display the number of words, lines and characters.

7.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out

7.5 SOURCE CODE:

Display the position of the string

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
        char a[50],b[50];
        char *ptr;
        clrscr();
        printf("\nEnter the string : ");
        scanf("%s",a);
        printf("\nEnter the substring : ");
        scanf("%s",b);
        ptr=strstr(a,b);
        if(ptr==NULL)
                printf("\n-1");
        else
                printf("Location of %s in %s = %d",b,a,ptr-a);
}
```

To count the lines, words & characters in a given text

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
        int i=0,wrd=0,lns=0,chr=0;
        char txt[100];
        char ch;
        printf("\nEnter the text( Enter $ at end ):");
        while( ch=getchar() )
        {
                txt[i++]=ch;
                if( ch == '$')
                         break;
        }
        txt[i]='\0';
        for(i=0;i<strlen(txt)-1;i++)
        {
          if(txt[i]==32 ||txt[i] == '\n'||txt[i] == ','||txt[i] == '.')
          {
                wrd++;
          }
          if(txt[i]=='\n')
          {
                lns++;
          }
        }
        chr = strlen(txt)-1;
        printf("no of words=%d\nNo of lines=%d\nNo of charecters=%d",wrd,lns,chr-lns);
}
```

7.6 INPUT/OUTPUT:

Display the position of the string

Enter the string :saikumar

```
Enter the substring :Kumar
position of Kumar in saikumar = 3
To count the lines, words & characters in a given text
Enter the text( Enter $ at end ):hai welcome to
c programming$
no of words=5
No of lines=2
No of characters=27
```

7.7 PRE-LAB VIVA QUESTIONS:

1 What is the difference between a string copy (*strcpy*) and a memory copy (*memcpy*) ? When should each be used ?

2 How can I remove the trailing spaces from a string ?

3 How can I remove the leading spaces from a string ?

7.8 LAB ASSIGNMENT:

1. Develop a C program that displays the position or index in the string S where the string T begins, or -1 if S does not contain T.
2. Read a string and delete the duplicate character in the string.
3. Find the count of occurrence of the character in the given string.
4. Accept two strings and compare one String with another String without using String Handling functions and find two strings are equal or not.
5. Write a program to count the lines, words and characters in a given text.
6. Read two strings, str1 as “goodday” and str2 as “noontime”. By using these strings generate a new string str3 as “goodnoon“. Now compare str3 with str1 to find the greatest ?

7.9 POST-LAB VIVA QUESTIONS:

1. What is the difference between printf() and puts() ?
2. Define pointer variable ?
3. What is use of the strcmp() function ?
4. What is the use of strcat() function ?

WEEK-8

8.1 OBJECTIVE:

1. To generate Pascal's triangle.
2. To generate Pyramid triangle.

8.2 RESOURCES:

centOS

8.3 PROGRAM LOGIC:

To generate pascal's triangle

```
1. Initialize m=0
2. Read n
3. If m<n goto step 5.if not goto step 12
4. initialize i=40-m
5. If i>0 is true do as follows. If not goto step 6
        i. print white space
        ii. decrement i
        iii. goto Step 6
6. Initialize j=0
7. If j=m do as follows. If not goto Step 9
        i. if(j==0||m==0)
        ii. Initialize b=1 if not b=b*(m-j+1)/j
        iii.Print white space, b .
        iv.Goto Step 8
8. increment j, goto Step 8
9. print new line control
10. increment m, goto step 3
```

To generate pyramid triangle

```
1. Read the value of N as the height of the pyramid.
2. Initialize i=0.
3. Display the current value of i each time and then increment it.
```

8.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out

8.5 SOURCE CODE:

To generate pascal's triangle

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int row,b,x,y,z;
        clrscr();
```

```
                b=1;
                y=0;
                        printf("\n Enter the number of rows for the pascal's tiangle:");
                scanf("%d",&row);
                while(y<row)
                {
                        for(x=40-(3*y);x>0;--x)
                        printf(" ");
                        for(z=0;z<=y;++z)
                        {
                                if((z==0)||(y==0))
                                b=1;
                                else
                                b=(b*(y-z+1))/z;
                                printf("%6d",b);
                        }
                        printf("\n");
                        ++y;
                }
                getch();
        }
```

To generate pyramid triangle

```
        #include<stdio.h>
        #include<conio.h>
        void gap(int n)
        {
                int i;
                for(i=0;i<n;i++)
                {
                        printf("  ");
                }
        }
        void main()
        {
                int i,j,n;
                printf("Enter the n:");
                scanf("%d",&n);
                for(i=1;i<n+1;i++)
                {
                        gap(n-i);
                        for(j=0;j<i;j++)
                        {
```

```
                printf("%d   ",i);
            }
            printf("\n");
        }}
```

8.6 INPUT/OUTPUT:

To generate pascal’s triangle

```
Enter the number of rows for the Pascal’s tiangle:5
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
```

To generate pyramid triangle

```
Enter the n:9
                1
              2   2
            3   3   3
          4   4   4   4
        5   5   5   5   5
      6   6   6   6   6   6
    7   7   7   7   7   7   7
  8   8   8   8   8   8   8   8
9   9   9   9   9   9   9   9   9
```

8.7 PRE-LAB VIVA QUESTIONS:

1 What is meant by Pascal’s triangle ?

2 What is a structure ?

3 Differentiate between array and structure ?

4 How do you access the member of a structure ?

8.8 LAB ASSIGNMENT:

1. Generate Pascal’s triangle
2. Construct a pyramid of numbers.
3. Print the multiplication table of a given number.
4. Sort the elements of an array in descending order.
5. Write a program to read an integer then display the value of that integer in decimal, octal, and hexadecimal notation ?

8.9 POST-LAB VIVA QUESTIONS:

1. What is meant by Pascal’s triangle ?
2. What is the use of dot operator in structures ?
3. Define union ?
4. What are embedded structures ?
5. Differentiate between address operator and dereferencing operator.

WEEK-9

9.1 OBJECTIVE:

A program to compute the sum of this geometric progression $1 + x + x^2 +$……………..$+x^n$.

9.2 RESOURCES:

centOS

9.3 PROGRAM LOGIC:

Sum of the geometric progression

1. Read the value of x and no. of terms n.
2. Initialize sum = 0 and i =0.
3. If n <= 0 || x <= 0 then display x and n values must be positive integers.
4. Otherwise find the sum = sum + pow(x, i).
5. Display the sum.

9.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out

9.5 SOURCE CODE:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
        int n,i,x,sum=0;
        printf("Enter the x and n");
        scanf("%d%d",&x,&n);
        if(n<=0||x<=0)
        {
                printf("the value must be positive integers");
        }
        else
        {
                for(i=0;i<=n;i++)
            {
                  sum=sum+pow(x,i);
            }
                printf("Sum=%d",sum);
        }
        getch();
}
```

9.6 INPUT/OUTPUT:

```
Sum of the geometric progression
Enter the x and n:5 3
Sum=156
```

9.7 PRE-LAB VIVA QUESTIONS:

1. What is a pointer ?
2. How do you declare a pointer variable
3. How do you use a pointer to a function ?
4. Difference between array and pointers ?

9.8 LAB ASSIGNMENT:

1. The first few numbers of the Lucas sequence which is a variation on the Fibonacci sequence are:

   1 3 4 7 11 18 29

   Formulate a program to generate the Lucas sequence.

1. Given a= 0, b = 1, and c = 1 are the first three numbers of some sequence. All other numbers in the sequence are generated from the sum of their three most recent predecessors. Compose a C-program to generate this sequence.
2. Write a C-program to evaluate the function sin(x) as defined by

$$\sin(x) = \frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

3. Write a C-program to evaluate the function cos(x) as defined by the infinite series expansion

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots$$

   The acceptable error from the computation is 10^{6}.

9.9 POST-LAB VIVA QUESTIONS:

1. How structure name is different from a structure variable ?
2. How can putchar() can be used to print a string ?
3. List out the rules for naming a variable ?
4. How is an array represented in memory ?
5. Define function ? Why are they needed ?

WEEK-10

10.1 OBJECTIVE:

1. To find the 2's compliments of a binary number
2. To convert a Roman numeral to its decimal equivalent

10.2 RESOURCES:

centOS

10.3 PROGRAM LOGIC:

2's compliment of a given binary number

1. Read no of bits in a binary number.
2. Read the binary number into an array.
3. Call the function twoscomplement.
4. Check individual digits of the binary number
   a. If the bit is 0 then convert to 1 and vice versa and print the 1's complement of the number.
5. Add 1 to 1's complement of the number
   a. If there is no carry then print the 2's complement.
   b. If there is a carry then subtract 2 and then print the 2's complement.

To convert Roman numeral to its decimal equivalent

1. Read a roman number.
2. Find the sum of their individual number weights.
3. If the weight of first number is less than second number then sum = sum -2*weight of first number.
4. Display the decimal equivalent of the roman number.

10.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out

10.5 PROCEDURE:

2's compliment of a given binary number

```
#include"stdio.h"
#include"conio.h"
void twoscomplement( int a[10],int n)
{
        int i,c=1;
        for(i=0;i<n;i++)
        {
                if(a[i]==0)
                        a[i]=1;
                else
                        a[i]=0;
        }
        printf("\n The 1's complement of the number is : ");
```

```
    for(i=0;i<n;i++)
        printf("%4d", a[i]);
    for(i=n-1;i>=0;i--)
    {
        a[i]=a[i]+c;
        if(a[i]==1)
        {
            c=0;
            break;
        }
        else
        {
            a[i]=a[i]-2;
            c=1;
        }
    }
    printf("\n");
    printf("\n The 2's Complement of the number is: ");
    if(c==1)
        printf("\n%4d",c);
    for(i=0;i<n;i++)
        printf("%4d", a[i]);
    getch();
}
void main()
{
    int n,i, a[10];
    clrscr();
    printf("\n number of bits in the binary number: ");
    scanf("%d", &n);
    printf("\n Enter the binary number: ");
    for(i=0;i<n;i++)
        scanf("%d", &a[i]);
    twoscomplement(a,n);
}
```

To convert Roman numeral to its decimal equivalent

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

```
            char arr[10];
            clrscr();
            printf("\nEnter a roman number : ");
            gets(arr);
            strupr(arr);
            for(i=0;i<strlen(arr);i++)
            {
                    switch(arr[i])
                    {
                            case 'M':tot[i]=1000;break;
                            case 'D':tot[i]=500;break;
                            case 'C':tot[i]=100;break;
                            case 'L':tot[i]=50;break;
                            case 'X':tot[i]=10;break;
                            case 'V':tot[i]=5;  break;
                            case 'I':tot[i]=1;break;
                            default:printf("wrong roman number");
                    }
                    sum+=tot[i];
            }
            for(i=0;i<strlen(arr)-1;i++)
            {
                    if(tot[i]<tot[i+1])
                    {
                            sum=sum-2*tot[i];
                    }
            }
            printf("\nDecimal equivalent of roman number = %d",sum);
            getch();
    }
```

10.6 INPUT/OUTPUT:

```
2's compliment of a given binary number
Number of bits in the binary number: 5
 Enter the binary number: 1 0 1 0 1
 The 1's complement of the number is :    0   1   0   1  0
The 2's Complement of the number is:     0   1   0   1   1


To convert Roman numeral to its decimal equivalent
Enter a roman number : xdcl
Decimal equivalent of roman number = 640
```

10.7 PRE-LAB VIVA QUESTIONS:

1. How to find two complement of a number ?

2. Explain the utility of typedef keyword in structures ?
3. What is the advantage of using structure ?
4. Explain with an example how structures are initialized ?
5. Difference between union and structure ?
6. Define enum data type ?

10.8 LAB ASSIGNMENT:

1. Compute area and volume of a cube.
2. Find square root of given number.
3. Convert uppercase character to lower case character and vice versa.
4. Convert the Roman numeral to its decimal equivalent.
5. Find the 2's compliment of a binary number.

10.9 POST-LAB VIVA QUESTIONS:

1. Expand ASCII ?
2. Define 2's complement ?
3. What are various operations performed on union ?
4. Explain the different modes in which a file can be opened ?
5. Under which circumstances does the fopen() fail ?

WEEK-11

11.1 OBJECTIVE:

Program that uses functions to perform the following operations

a. Reading a complex number
b. Writing a complex number
c. Addition of 2 complex numbers
d. Multiplication of 2 complex numbers

11.2 RESOURCES:

centOS

11.3 PROGRAM LOGIC:

Arithmetic operations using complex numbers

1. Define a structure having to members one is real (r) and other is imaginary (i)
2. Create two structure variable and read the real and imaginary parts of two complex numbers.
3. Compute the addition of complex numbers by adding real part with real and imaginary with imaginary. (c3. R = c1. R + c2. R and c3. I = c1. I + c2. I)
4. Compute the multiplication of complex numbers by using the formula
   c3. R = (c1. R * c2. R) – (c1.I * c2.I)
   c3. I = (c1.R * c2. I) – (c1.I * c2.R)
5. Display the complex numbers.

11.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out

11.5 SOURCE CODE:

```
# include <stdio.h>
struct complex   // user defined data type
{
        int x;
        int y;
}; // variable
void add(struct complex c1, struct complex c2)
{
        struct complex c3;
        c3.x = c1.x + c2.x;
        c3.y = c1.y +c2.y;
        printf("\n sum c3 = %d + %d i",c3.x,c3.y);
}
void sub(struct complex c1, struct complex c2)
{
        struct complex c3;
        c3.x = c1.x - c2.x;
```

```
            c3.y = c1.y -c2.y;
            printf("\n diff c3 = %d + %d i",c3.x,c3.y);
    }
    void mul(struct complex c1, struct complex c2)
    {
            struct complex c3;
            c3.x =   (c1.x * c2.x)- (c1.y *c2.y);
            c3.y =   (c1.x * c2.y) + (c1.y * c2.x);
            printf("\n mul c3 = %d + %d i",c3.x,c3.y);
    }

    main()
    {
            struct complex c1,c2;
            printf("\n Enter Real and Imaginary part of c1 : ");
            scanf("%d%d",&c1.x,&c1.y);
            printf("\n Enter Real and Imaginary part of c2 : ");
            scanf("%d%d",&c2.x,&c2.y);

            add( c1,c2);
            sub(c1,c2);
            mul(c1,c2);
    getch();
    }
```

11.6 INPUT/OUTPUT:

```
Arithmetic operations using complex numbers
Enter Real and Imaginary part of c1 : 1 3
 Enter Real and Imaginary part of c2 : 2 3
 sum c3 = 3 + 6 i
 diff c3 = -1 + 0 i
 mul c3 = -7 + 9 i
```

11.7 PRE-LAB VIVA QUESTIONS:

1. What do you understand by EOF ?
2. Define the term stream ?
3. Explain the difference between a NULL pointer and Void pointer ?
4. Differentiate between ptr++ and *ptr++ ?

11.8 LAB ASSIGNMENT:

1. Formulate a Program to Store Information (name, roll and marks) of a Student Using Structure .
2. Formulate a Program to Add Two Distances (in inch-feet) System Using Structures .
3. Formulate a Program to Add Two Complex Numbers by Passing Structure to a Function .
4. Formulate a Program to Store Information of 10 Students Using Structure.
5. Formulate a Program to Store Information Using Structures for n Elements Dynamically.

6. Formulate a program to declare pointers as members of the structure and display the contents of that structure. Define a structure object student with three fields: name, age and height ?
7. Write a program to read n records of students and find out how many of them have passed. The fields are student's roll-no, name, marks and result. Evaluate the result as follows.

   if marks>35 then Result="pass" else "Fail"
8. A marketing company is having 50 employees and it maintains employee records in terms of their empid, empname, desg, salary, quantity, salesamount. The company gives 10% hike in salary to the employees if their salesamount is more than 50000/-. Display the employee records that got hike in salary.
9. IARE College is maintaining student attendance records by storing rollno, stdname, attendance percentage in 5 different subjects. Write a C program to find the average attendance percentage and print the following
   a) If attendance percentage >=75 then print student is eligible for writing final exam.
   b) If attendance percentage >= 65 and <75 then print student is in condonation list.
   c) Otherwise not eligible for writing exams.
10. Create a Books structure having book_id, book_name, title, author, and price. Pass the structure as a function argument and print the details of the book.

11.9 POST-LAB VIVA QUESTIONS:

1. What is use of <math.h> header file ?
2. Is it possible to create an array of structure ?
3. Differentiate between a structure and union with respect to allocation of memory by the compiler. Give an example of each ?
4. What are Bitfields ? What are its advantages ?
5. What do you mean by self referential structure and give one example ?

WEEK-12

12.1 OBJECTIVE:

To perform the following operations on file i.e copying the file, reverse the characters .

12.2 RESOURCES:

centOS

12.3 PROGRAM LOGIC:

Copy the contents from one file to another

```
1. Start
2. read command line arguments
3. check if no of arguments =3 or not. If not print invalid no of arguments
4. open source file in read mode
5. if  NULL pointer, then print source file cannot be open
6. open destination file in write mode
7. if NULL pointer, then print destination file cannot be open
8. read a character from source file and write to destination file until EOF
9. Close source file and destination file
10. Stop
```

Reverse n character in a file

```
1. Start
2. read the command line arguments
3. check if arguments=3 or not, If not print invalid no of arguments
4. open source file in read mode
5. if NULL pointer, then print file cannot be open
6. Store no of chars to reverse in k
        K= *argv[2]-48
7. read the item from file stream using fread
8. Store chars from last position to initial position in another string(temp)
9. print the temp string
10. Stop
```

12.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out

12.5 SOURCE CODE:

Copy the contents from one file to another

```
#include <stdio.h>
#include <conio.h>
#include <process.h>
void main(int argc, char *argv[])
{
        FILE *fs,*ft;
```

```
    char ch;
    clrscr();
    if(argc!=3)
    {
        puts("Invalid number of arguments.");
        exit(0);
    }
    fs = fopen(argv[1],"r");
    if(fs==NULL)
    {
    puts("Source file cannot be opened.");
    exit(0);
    }
    ft = fopen(argv[2],"w");
    if (ft==NULL) // check the condition if the file pointer is NULL or not
    {
        puts("Target file cannot be opened.");
        fclose(fs);
        exit(0);
    }
    while(1)
    {
        ch=fgetc(fs);
        if (ch==EOF) // check the condition if the file is end or not
        break;
        else
        fputc(ch,ft);
    }
    fclose(fs);
    fclose(ft);
    getch();
}
```

<u>Reverse n character in a file</u>

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <process.h>
void main(int argc, char *argv[])
{
    char a[15];
    char s[20];
    char n;
```

```
        int k;
        int j=0;
        int i;
        int len;
        FILE *fp;
        if(argc!=3)
        {
                puts("Improper number of arguments.");
                exit(0);
        }
        fp = fopen(argv[1],"r");
        if(fp == NULL)
        {
                puts("File cannot be opened.");
        exit(0);
        }
        k=*argv[2]-48;
        n = fread(a,1,k,fp);
        a[n]='\0';
        len=strlen(a);
        for(i=len-1;i>=0;i--)
        {
                s[j]=a[i];
                printf("%c",s[j]);
                j=j+1;
        }
        s[j+1]='\0';
        getch();
}
```

12.6 INPUT/OUTPUT:

```
Copy the contents from one file to another
source.c
      this is source text
ouput.c
Command line arguments
source.c ouput.c
source.c
      this is source text
ouput.c
      this is source text

Reverse n character in a file
```

```
source.c
          this is source
ouput.c
Command line arguments
source.c ouput.c
source.c
          this is source
          ecruos si siht
```

12.7 PRE-LAB VIVA QUESTIONS:

1. What is file ?
2. What are the various operations performed on the file ?
3. What is the use of file pointer ?
4. List out the file handling functions ?
5. What is the use of fseek() function ?

12.8 LAB ASSIGNMENT:

1. Copy the contents from one file to another file.
2. Reverse n character in a file.
3. To count the number of lines and words in a file
4. Write employees details in a file called employee data. Then read the record of the nth employee and calculate his salary ?
5. Read the text file containing some paragraph. Use fseek() and read the text after skipping ‘n’ characters from beginning of the file ?

12.9 POST-LAB VIVA QUESTIONS:

1. What is use of the fflush() function ?
2. List out some of the math functions ?
3. Explain I/O Statements ?
4. Describe types of files with an example ?
5. What is the task performed by fseek() function ?

WEEK-13

13.1 OBJECTIVE:

1. Displaying the contents of file.
2. Merging of two files into a third file.

13.2 RESOURCES:

centOS

13.3PROGRAM LOGIC:

Displaying the contents of file.

```
1. Start
2. read command line arguments
3. check if no of arguments =3 or not. If not print invalid no of arguments
4. open source file in read mode
5. if  NULL pointer, then print source file cannot be open
6. open destination file in write mode print a content of file
7. if NULL pointer, then print destination file cannot be open
8. read a character from source file and write to destination file until EOF
9. Close source file and destination file
10. Stop
```

Merging of two files into a third file.

```
1. Start
2. read command line arguments
3. check if no of arguments =3 or not. If not print invalid no of arguments
4. open source file 1 in read mode
5. open source file 2 in read mode
6. open destination file3 in write mode
7. write the file1 and file2 content into file3
8. Stop
```

13.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and: wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: . / a.out

13.5 SOURCE CODE:

Displaying the contents of file:

```
#include<stdio.h>
#include<conio.h>
void main()
{
 FILE *fp;
 char ch;
 clrscr();

/* open the file in read mode*/
```

```
 fp = fopen("text.dat","r");
 if(fp==NULL)
            printf("No such file\n");
 else
 {
  printf("File contents are:\n");
  while(!feof(fp))
  {
            ch = fgetc(fp);
            putchar(ch);
    }
  fclose(fp);
  }
  getch();
}
```

Merging of two files into a third file

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

int main()
{
FILE *fs1, *fs2, *ft;

char ch, file1[20], file2[20], file3[20];

printf("Enter name of first file\n");
gets(file1);

printf("Enter name of second file\n");
gets(file2);

printf("Enter name of file which will store contents of two files\n");
gets(file3);

fs1 = fopen(file1,"r");
fs2 = fopen(file2,"r");

if( fs1 == NULL || fs2 == NULL )
{
printf("Error in files");
}
else
{
   ft = fopen(file3,"w");

  if( ft == NULL )
 {
   printf("Error ");

  }
else
{
while( ( ch = fgetc(fs1) ) != EOF )
fputc(ch,ft);


while( ( ch = fgetc(fs2) ) != EOF )
fputc(ch,ft);

printf("Two files were merged into %s file successfully.\n",file3);

fclose(fs1);
fclose(fs2);
fclose(ft);
}
}

return 0;
}
```

13.6 INPUT/OUTPUT:

```
Enter name of first file1
Enter name of second  file2
Enter name of file which will store contents of two files
Two files were merged into %s file successfully
```

13.7 PRE-LAB VIVA QUESTIONS:

1 .What is a FILE?

2. What is the purpose of the fopen() function?

3. What is the purpose of the fclose() function? Is it mandatory to use this in a program that processes a data file?

4 . What is the difference between a binary file and a text file in C?

5 . Compare fscanf() and fread() functions?

13.8 LAB ASSIGNMENT:

1. Write a code that will read a line of characters (terminated by a \n)from a text file into a character array called buffer. NULL terminates the buffer upon reading a \n.
2. Write a program to copy one existing file into another named file.
3. Write a program that opens a file and counts the number of characters .The program should print out the lines where they differ?
4. Write a program that opens an existing text file and copies it to a new text file with all lowercase letters changed to capital letters and all other characters unchanged
5. Write an interactive C program that will maintain a list roll, name and total marks of students. select .any of the following .
   - i) Add a new record.
   - ii) Delete a record.
   - iii) Modify a record
   - iv) Retrieve and display an entire record for a given roll or name
   - v) Display all records.
   - vi) End of computation

13.9 POST-LAB VIVA QUESTIONS:

1.What is stream? describe two different methods of creating a stream –oriented data file.

2 . Write a program to compare two files and print out the lines where they differ?
3. What is EOF? When is EOF used?

4 .What are the three general methods of the access?

5. What is the difference between a text–mode stream and a binary–mode stream

WEEK-14

14.1 OBJECTIVE:

1. Non recursive functions to perform linear search for a Key value in a given list.
2. Non recursive functions to perform binary search for a key value in a given list.

14.2 RESOURCES:

centOS

14.3 PROGRAM LOGIC:

Linear search:

1. Read search element.
2. Call function linear search function by passing N value, array and search element.
3. If a[i] ==k, return i value, else return -1, returned value is stored in pos.
4. If pos ==-1 print element not found, else print pos+1 value.

Binary search:

1. Read search element.
2. Call binary search function with values N, array, and data.
3. If low is less than high, making mid value as mean of low and high.
4. If a [mid] ==data, make flag=1 and break, else if data is less than a[mid] make high=mid-1 else low=mid+1.
5. If flag ==1, print data found at mid+1, else not found.

14.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

14.5 SOURCE CODE:

Linear search:

```
#include<stdio.h>
#include<conio.h>
int linear_search(int n,int a[20],int k)
{
    int i=0;
    for(i=0;i<n;i++)
    {
            if(a[i]==k)
            return i;
    }
    return -1;
}
void main()
{
    int i,a[20],n,k,pos;
    clrscr();
    printf("Enter no of elements:");
    scanf("%d",&n);
    printf("Enter elements:");
    for(i=0;i<n;i++)
            scanf("%d",&a[i]);
    printf("Enter search element:");
    scanf("%d",&k);
    pos=linear_search(n,a,k);
    if(pos==-1)
            printf("%d is not found",k);
    else
            printf("%d is found at position %d",k,pos+1);
    getch();
}
```

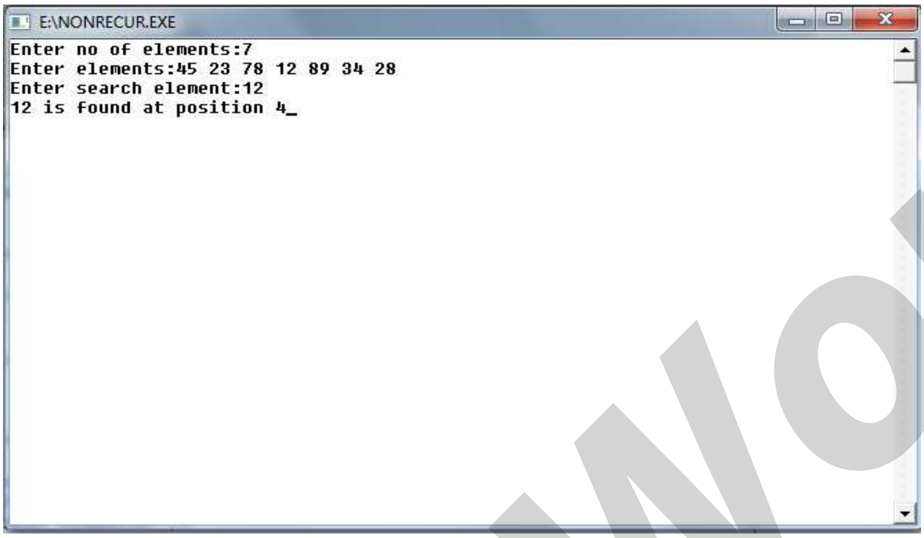
Binary search:

```
#include<stdio.h>
#include<conio.h>
void binary_search(int n,int a[20],int data)
{
    int mid,flag=0;
    int low=0,high;
    high=n-1;
    while(low<=high)
    {
            mid=(low+high)/2;
            if(a[mid]==data)
            {
                 flag=1;
                 break;
            }
            else
            {
                    if(data<a[mid])
                            high=mid-1;
                    else
                            low=mid+1;
            }
    }
    if(flag==1)
            printf("\nData found at %d location",mid+1);
    else
            printf("Not found");
}

void main()
{
    int i,a[20],n,k;
    clrscr();
    printf("Enter no of elements:");
    scanf("%d",&n);
    printf("Enter elements:");
for(i=0;i<n;i++)
        scanf("%d",&a[i]);
printf("Enter search element:");
scanf("%d",&k);
binary_search(n,a,k);
}
```

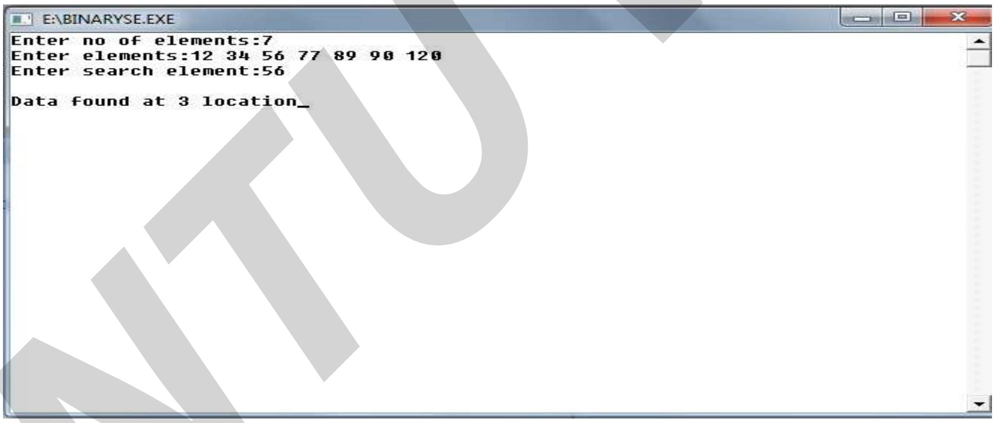
14.6 INPUT/OUTPUT:

Linear search:



```
Enter no of elements:7
Enter elements:45 23 78 12 89 34 28
Enter search element:12
12 is found at position 4_
```

Binary search:



```
Enter no of elements:7
Enter elements:12 34 56 77 89 90 120
Enter search element:56

Data found at 3 location_
```

14.7 PRE-LAB VIVA QUESTIONS:

1. Define a Data Structure?
2. What is a Linear Data Structure?
3. What is a Non Linear Data Structure?
4. What is meant by searching?
5. What is the basic idea of Linear Search?
6. What is the basic idea of Binary Search?
7. What do you mean by Fibonacci Search?

14.8 LAB ASSIGNMENT:

1. Compose a program that uses both recursive and non recursive functions to perform linear search for a Key value in a given list.
2. Write a program that uses both recursive and non recursive functions to perform binary search for a Key value in a given list.
3. Formulate a program that uses both recursive and non recursive functions to perform Fibonacci search for a Key value in a given list.
4. Write a program to find the occurrence of a given character from a string using linear search.

Compose a program to search a book title from library using binary search

14.9 POST-LAB VIVA QUESTIONS:

1. What is the time complexity of Linear Search?
2. What is the time complexity of Binary Search?
3. What is the time complexity of Fibonacci Search?
4. What is the advantage of binary search over linear search?
5. Which concept is used in binary search?
6. What is the minimum requirement for binary search?
7. How Fibonacci search works?

WEEK-15

15.1 OBJECTIVE:

1. To implements the Selection sort method to sort a given array of integers in ascending order.

2 .To implements the Bubble sort method to sort a given list of names in ascending order.

15.2 RESOURCES:

centOS

15.3 PROGRAM LOGIC:

Selection sort

1. Start with the result as the first element of the input.
2. Loop over the input until it is empty, "removing" the first remaining (leftmost) element.
3. Compare the removed element against the current result, starting from the highest (rightmost) element, and working left towards the lowest element.
4. If the removed input element is lower than the current result element, copy that value into the following element to make room for the new element below, and repeat with the next lowest result element.
5. Otherwise, the new element is in the correct location; save it in the cell left by copying the last examined result up, and start again from (2) with the next input element.

Bubble sort

1. Start with the result as the first element of the input.
2. Loop over the input until it is empty, "removing" the first remaining (leftmost) element.
3. Compare the removed element against the current result, starting from the highest (rightmost) element, and working left towards the lowest element.
4. If the removed input element is lower than the current result element, copy that value into the following element to make room for the new element below, and repeat with the next lowest result element.

Otherwise, the new element is in the correct location; save it in the cell left by copying the last examined result up, and start again from (2) with the next input element

15.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

15.5 SOURCE CODE:

Selection sort

```
#include <stdio.h>
int main()
{
 int array[100], n, c, d, swap;
  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 = 0 ; c < ( n - 1 ); c++)
 {
   for (d = 0 ; d < n - c - 1; d++)
   {
     if (array[d] > array[d+1]) /* For decreasing order use < */
     {
       swap       = array[d];
       array[d]   = array[d+1];
       array[d+1] = swap;
     }
   }
 }
 printf("Sorted list in ascending order:\n");
for ( c = 0 ; c < n ; c++ )
    printf("%d\n", array[c]);
 return 0;
}
```

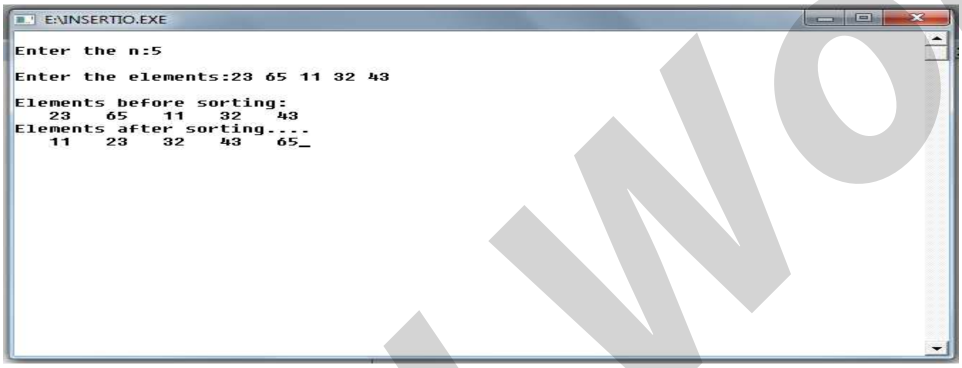
Bubble sort

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

}
printf("After sorting, the element are\n");
for(i=0;i<n;i++)
        printf("%4d",a[i]);
getch();
return 0;
}
```

15.6 INPUT/OUTPUT:

Selection sort:

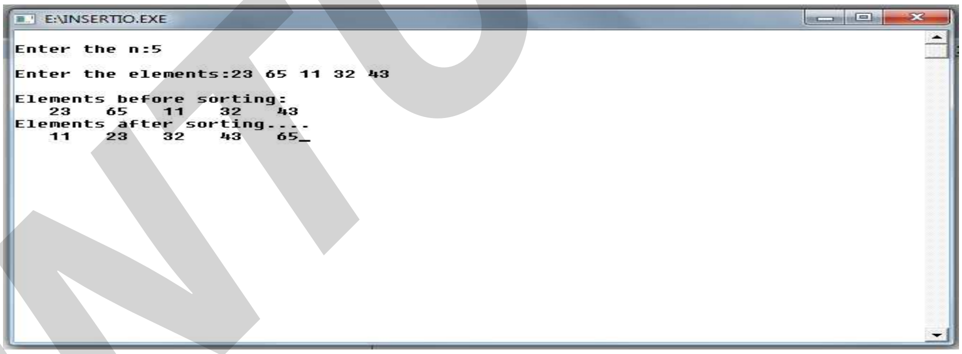


```
Enter the n:5

Enter the elements:23 65 11 32 43

Elements before sorting:
   23   65   11   32   43
Elements after sorting....
   11   23   32   43   65_
```

Bubble sort



```
Enter the n:5

Enter the elements:23 65 11 32 43

Elements before sorting:
   23   65   11   32   43
Elements after sorting....
   11   23   32   43   65_
```

15.7 PRE-LAB VIVA QUESTIONS:

1. What is meant by Sorting?
2. What do you mean by Internal Sorting?
3. What do you mean by External Sorting?
4. What is the basic idea of Bubble Sort?
5. How to sort the elements using Insertion Sort?
6. What is the other name of Quick Sort?

15.8 LAB ASSIGNMENT:

1. Formulate a program that implement Bubble sort, to sort a given list of integers in ascending order.
2. Write a program that implement Quick sort, to sort a given list of integers in ascending order.
3. Compose a program that implement Insertion sort, to sort a given list of integers in ascending order.
4. Write a program that implement Selection sort, to sort a given list of integers in ascending order.
5. Formulate a program to sort N names using selection sort.
6. Write a program to sort N employee records based on their salary using insertion sort.

15.9 POST-LAB VIVA QUESTIONS:

1. What is the time complexity of Bubble Sort?
2. What is the time complexity of Insertion Sort?
3. Quick Sort works by which strategy?
4. What is the time complexity of Quick Sort?
5. How is the procedure used to sort the elements using Selection Sort?
6. What is the time complexity of Selection Sort?

WEEK-16

16.1 OBJECTIVE:

To create a singly linked list and traversing operations on a singly linked list

16.2 RESOURCES:

centOS

16.3 PROGRAM LOGIC:

1. Creating a structure with one data item and a next pointer, which will be pointing to next node of the list. This is called as self-referential structure.
2. Initialize the start pointer to be NULL.

Insertion of a Node:

. The new node can then be inserted at three different places namely:

1. Inserting a node at the beginning.
2. Inserting a node at the end.
3. Inserting a node at intermediate position

Inserting a node at the beginning:

The following steps are to be followed to insert a new node at the beginning of the list:

1. Get the new node using getnode().
   ```
   newnode = getnode();
   ```
2. If the list is empty then *start = newnode.*
3. If the list is not empty, follow the steps given below:
   ```
   newnode -> next = start;
   start = newnode;
   ```

Inserting a node at the end:

1. The following steps are followed to insert a new node at the end of the list:
2. Get the new node using getnode()
   ```
   newnode = getnode();
   ```
3. If the list is empty then *start = newnode*.
4. If the list is not empty follow the steps given below:
   ```
   temp = start;
   while(temp -> next != NULL)
           temp = temp -> next;
   temp -> next = newnode;
   ```

Inserting a node at intermediate position:

1. The following steps are followed, to insert a new node in an intermediate position in the list:
2. Get the new node using getnode().
   ```
   newnode = getnode();
   ```
3. Ensure that the specified position is in between first node and last node. If not, specified position is invalid. This is done by countnode() function.
4. Store the starting address (which is in start pointer) in temp and prev pointers. Then traverse the temp pointer upto the specified position followed by prev pointer.
5. After reaching the specified position, follow the steps given below:
   ```
   prev -> next = newnode;

   newnode -> next = temp;
   ```

Deletion of a node:

A node can be deleted from the list from three different places namely.

1. Deleting a node at the beginning.
2. Deleting a node at the end.
3. Deleting a node at intermediate position.

Deleting a node at the beginning:

The following steps are followed, to delete a node at the beginning of the list:

1. If list is empty then display ‘Empty List’ message.
2. If the list is not empty, follow the steps given below:

```
temp = start;
start = start -> next;
free(temp);
```

Deleting a node at the end:

1. The following steps are followed to delete a node at the end of the list:
2. If list is empty then display ‘Empty List’ message.
3. If the list is not empty, follow the steps given below:

```
temp = prev = start;
while(temp -> next != NULL)
{
    prev = temp;
    temp = temp -> next;
}
prev -> next = NULL;

free(temp);
```

Deleting a node at Intermediate position:

The following steps are followed, to delete a node from an intermediate position in the list (List must contain more than two node).

1. If list is empty then display ‘Empty List’ message
2. If the list is not empty, follow the steps given below.

```
if(pos > 1 && pos < nodectr)
{
        temp = prev = start;
        ctr = 1;
        while(ctr < pos)
                                                        {
            prev = temp;
            temp = temp -> next;
            ctr++;
        }
        prev -> next = temp -> next;
        free(temp);
        printf("\n node deleted..");
}
```

Traversal and displaying a list (Left to Right):

Traversing a list involves the following steps:

1. Assign the address of start pointer to a temp pointer.
2. Display the information from the data field of each node.
3.

16.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.

2. Compile: gcc x.c.
3. Execute: ./ a.out

16.5 SOURCE CODE:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct linklist
{
	int data;
	struct linklist *next;
};
typedef struct linklist node;
node *start=NULL;
int menu()
{
	int ch;
	printf("\n\t *****IMPLIMENTATION OF SINGLE LINKED LIST*****");
	printf("\n\t ---------------------------------------\n");
	printf("\n\t 1.Create list");
	printf("\n\t 2.Traverse the list(left to right)");
	printf("\n\t 3.Traverse the list(right to left)");
	printf("\n\t 4.Number of nodes");
	printf("\n\t 5.Insertion at Begining");
	printf("\n\t 6.Insertion at End");
	printf("\n\t 7.Insertion at Middle");
	printf("\n\t 8.Deletion at Beginning");
	printf("\n\t 9.Deletion at End");
	printf("\n\t 10.Deletion at Middle");
	printf("\nEnter your choice:");
	scanf("%d",&ch);
	return ch;
}
node* getnode()
{
	node *newnode;
	newnode=(node*)malloc(sizeof(node));
	printf("Enter data:\n");
	scanf("%d",&newnode->data);
	newnode->next=NULL;
	return newnode;
}
int countnode(node*start)
{
	if(start==NULL)
		return 0;
	else
		return 1+countnode(start->next);
}

void createlist(int n)
{
	int i;
	node *newnode;
	node *temp;
```

```
    for(i=0;i<n;i++)
    {
            newnode=getnode();
            if(start==NULL)
            {
                    start=newnode;
            }
            else
            {
                    temp=start;
                    while(temp->next!=NULL)
                            temp=temp->next;
                    temp->next=newnode;
            }
    }
}
void traverse()
{
    node *temp;
    temp=start;
    printf("The contents of the list(left to right)\n");
    if(start==NULL)
            {
                    printf("\n Empty list");
                    return;
            }
    else
    {
            while(temp!=NULL)
            {
                    printf("%d\t",temp->data);
                    temp=temp->next;
            }
    }
    printf("X");
}
void rev_traverse(node *start)
{
    if(start==NULL)
            return;
    else
    {
            rev_traverse(start->next);
            printf("%d\t",start->data);
    }
}
void insert_at_beg()
{
    node *newnode;
    newnode=getnode();
    if(start==NULL)
    {
            start=newnode;
    }
    newnode->next=start;
```

```
        start=newnode;
}
void insert_at_end()
{
        node *newnode,*temp;
        newnode=getnode();
        if(start==NULL)
        {
                start=newnode;
        }
        else
        {
                temp=start;
                while(temp->next!=NULL)
                        temp=temp->next;
                temp->next=newnode;
        }

}
void insert_at_mid()
{
        node *newnode,*pre,*temp;
        int pos,ctr=1,nodectr;
        printf("Enter position:");
        scanf("%d",&pos);
        nodectr=countnode(start);
        if(pos>1 && pos<nodectr)
        {
                newnode=getnode();
                temp=pre=start;
                while(ctr<pos)
                {
                        pre=temp;
                        temp=temp->next;
                        ctr++;
                }
                pre->next=newnode;
                newnode->next=temp;
        }
        else
        {
                        printf("\nNot a middle position");
        }
}
void del_at_beg()
{
        node *temp;
        if(start==NULL)
        {
                printf("List is empty");
                return;
        }
        else
        {
                temp=start;
```

```
            start=temp->next;
            free(temp);
            printf("Node is deleted");
    }
}
void del_at_end()
{
    node *pre,*temp;
    if(start==NULL)
    {
            printf("List is empty");
            return;
    }
    else
    {
            temp=start;
            while(temp->next!=NULL)
            {
                    pre=temp;
                    temp=temp->next;
            }
            pre->next=NULL;
            free(temp);
            printf("\Node deleted");
    }
}
void del_at_mid()
{
    int pos,ctr=1,nodectr;
    node *temp,*pre;
    nodectr=countnode(start);
    if(start==NULL)
    {
            printf("List is empty");
            return;
    }
    else
    {
            printf("Enter position:");
            scanf("%d",&pos);
            if(pos>1 && pos<nodectr)
            {
                    pre=temp=start;
                    while(ctr<pos)
                    {
                            pre=temp;
                            temp=temp->next;
                            ctr++;
                    }
                    pre->next=temp->next;
                    free(temp);
            }
            else
            printf("Not a mid position");
    }
```

```
}
void main(void)
{
    int ch,n;
    clrscr();
    while(1)
    {
            ch=menu();
            switch(ch)
            {
                    case 1:  if(start==NULL)
                             {
                                     printf("Enter the number of nodes you want to create:");
                                     scanf("%d",&n);
                                     createlist(n);
                                     printf("List is created");
                                     break;
                             }
                             else
                             {
                                     printf("List is already created:");
                                     break;
                             }
                    case 2:traverse();break;
                    case 3:  printf("The contents of the list(left to right):\n");
                             rev_traverse(start);
                             printf("X");
                             break;
                    case 4:  printf("Number of nodes:%d",countnode(start));
                             break;
                    case 5:  insert_at_beg();
                             break;
                    case 6:  insert_at_end();
                             break;
                    case 7:  insert_at_mid();
                             break;
                    case 8:  del_at_beg();
                             break;
                    case 9:  del_at_end();
                             break;
                    case 10:del_at_mid();
                             break;
                    case 11:exit(0);
            }
    }
}
```

16.6 INPUT/OUTPUT:

```
*****IMPLIMENTATION OF LINKED LIST*****
                        ---------------------------------------

                        1.Create list
                        2.Traverse the list(left to right)
                        3.Traverse the list(right to left)
                        4.Number of nodes
```

```
                                5.Insertion at Begining
                                6.Insertion at End
                                7.Insertion at Middle
                                8.Deletion at Beginning
                                9.Deletion at End
                                10.Deletion at Middle
Enter your choice:1
Enter the number of nodes you want to create:5
Enter data: 1
Enter data: 2
Enter data: 3
Enter data: 4
Enter data: 5
List is created
                                *****IMPLIMENTATION OF LINKED LIST*****
                                ---------------------------------------

        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:2
The contents of the list(left to right)
1       2       3       4       5       X
        *****IMPLIMENTATION OF LINKED LIST*****
                                ---------------------------------------

        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:3
The contents of the list(left to right):
5       4       3       2       1       X
        *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

       1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:4
Number of nodes:5
                                *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------
        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
                                10.Deletion at Middle
Enter your choice:5
Enter data:
0

        *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

       1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:2
The contents of the list(left to right)
0       1       2       3       4       5       X
        *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
       10.Deletion at Middle
Enter your choice:6
Enter data:
6

                                *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:2
The contents of the list(left to right)
```

```
0     1     2     3     4     5     6     X
      *****IMPLIMENTATION OF LINKED LIST*****
      ---------------------------------------

      1.Create list
      2.Traverse the list(left to right)
      3.Traverse the list(right to left)
      4.Number of nodes
      5.Insertion at Begining
      6.Insertion at End
      7.Insertion at Middle
      8.Deletion at Beginning
      9.Deletion at End
      10.Deletion at Middle
Enter your choice:7
Enter position:4
Enter data:
7

      *****IMPLIMENTATION OF LINKED LIST*****
      ---------------------------------------

      1.Create list
      2.Traverse the list(left to right)
      3.Traverse the list(right to left)
      4.Number of nodes
      5.Insertion at Begining
      6.Insertion at End
      7.Insertion at Middle
      8.Deletion at Beginning
      9.Deletion at End
      10.Deletion at Middle
Enter your choice:2
The contents of the list(left to right)
0     1     2     7     3     4     5     6     X
      *****IMPLIMENTATION OF LINKED LIST*****
      ---------------------------------------

      1.Create list
      2.Traverse the list(left to right)
      3.Traverse the list(right to left)
      4.Number of nodes
      5.Insertion at Begining
      6.Insertion at End
      7.Insertion at Middle
      8.Deletion at Beginning
      9.Deletion at End
      10.Deletion at Middle
Enter your choice:8
Node is deleted
      *****IMPLIMENTATION OF LINKED LIST*****
                              ---------------------------------------

      1.Create list
      2.Traverse the list(left to right)
      3.Traverse the list(right to left)
      4.Number of nodes
      5.Insertion at Begining
      6.Insertion at End
      7.Insertion at Middle
      8.Deletion at Beginning
      9.Deletion at End
      10.Deletion at Middle
```

```
Enter your choice:2
The contents of the list(left to right)
1       2       7       3       4       5       6       X
        *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

                                      1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:9
Node deleted
                                      *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:2
The contents of the list(left to right)
1       2       7       3       4       5       X
        *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
Enter your choice:10
Enter position:3

        *****IMPLIMENTATION OF LINKED LIST*****
        ---------------------------------------

        1.Create list
        2.Traverse the list(left to right)
        3.Traverse the list(right to left)
        4.Number of nodes
        5.Insertion at Begining
        6.Insertion at End
        7.Insertion at Middle
        8.Deletion at Beginning
        9.Deletion at End
        10.Deletion at Middle
```

```
Enter your choice:2
The contents of the list(left to right)
1     2     3     4     5     X
                              *****IMPLIMENTATION OF LINKED LIST*****
      ------------------------------------

      1.Create list
      2.Traverse the list(left to right)
      3.Traverse the list(right to left)
      4.Number of nodes
      5.Insertion at Begining
      6.Insertion at End
      7.Insertion at Middle
      8.Deletion at Beginning
      9.Deletion at End
      10.Deletion at Middle
Enter your choice:11
```

16.7 PRE-LAB VIVA QUESTIONS:

1. What is linked list?
2. What type of memory allocation is used in linked list?
3. How many self referential pointers are used in single linked list?
4. What is double linked list?
5. Which node contains NULL pointer in a single linked list?
6. How many nodes you can have in a single linked list?

16.8 LAB ASSIGNMENT:

Formulate a program to create a singly linked list and perform insertion, deletion and traversing operations on a singly linked list.

1. Write a program to merge two linked list?
2. Compose a program to print odd nodes of a linked list?
3. Write a program to divide the linked list into two parts into odd and even list?
4. Formulate a program to convert a single linked to circular linked list?

16.9 POST-LAB VIVA QUESTIONS:

1. What is the time complexity to insert a node at the beginning of linked list?
2. What is the time complexity to traverse a linked list?
3. How many modifications are required to delete a node at the beginning?
4. How many modifications are required to insert a node in the middle of the linked list?

WEEK-17

17.1 OBJECTIVE:

Program that evaluate implement stack

17.2 RESOURCES:

centOS

17.3 PROGRAM LOGIC:

1. **STACK:** Stack is a linear data structure which works under the principle of **last in first out. Basic operations:** push, pop, display.
2. **PUSH**: if (newnode==NULL), display **Stack overflow**. Otherwise reading the data and making stack [top] =data and incrementing the top value by doing top++.
3. **Pop:** if (top == NULL), display **Stack underflow**. Otherwise printing the element at the top of the stack and decrementing the top value by doing the top.
4. **DISPLAY**: IF (top == NULL), display **Stack is empty**. Otherwise printing the elements in the stack from stack [0] to stack [top].

17.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

17.5 SOURCE CODE:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct stack
{
    int data;
    struct stack *next;
};
typedef struct stack node;
node *start = NULL;
node *top = NULL;
node* getnode()
{
    node *temp;
    temp=(node *)malloc(sizeof(node));
    printf("\n enter data : ");
    scanf("%d",&temp -> data);
    temp -> next=NULL;
    return temp;
}
void push(node *newnode)
{
    node *temp;
            if(newnode==NULL)
            {
                    Printf ("\n stack overflow... ");
                    return;
            }
            if(start == NULL)
            {
                    start = newnode;
            top = newnode;
        }
        else
        {
            temp = start;
            while(temp -> next!= NULL)
                temp = temp -> next;
            temp -> next =newnode;
            top = newnode;
        }
        Printf ("\n\n\t data pushed into stack...");

}
void pop()
{
    node *temp;
    if(top == NULL)
    {
         printf("\n\n\t stack overflow...");
         return;
    }
    temp = start;
    if(start -> next == NULL)
    {
        printf("\n\n\t popped element is %d",top -> data);
        start = NULL;
        free(top);
        top = NULL;
    }
    else
    {
        while(temp -> next!=top)
        {
            temp = temp -> next;
        }
        temp -> next = NULL;
        printf("\n\n\t popped element is %d", top -> data);
        free(top);
        top = temp;
    }

}
void display()
{
    node *temp;
    if(top == NULL)
    {
        printf("\n\n\t stack is empty...");
    }
    else
    {
        temp = start;
        Printf ("\n\n\t elements in the stack..\n");
        printf("%5d",temp -> data);
        while(temp!= top)
```

```
                    {
                            temp = temp -> next;
                            printf("%5d",temp -> data);
                    }
            }
    }
    int menu()
    {
            int ch;
            printf("\n\t STACK OPERATIONS USING POINTERS : ");
            printf("\n ----******----- ");
            printf("\n 1. Push ");
            Printf ("\n 2. Pop ");
            printf("\n 3. Display ");
            printf("\n 4. Quit");
            printf("\n enter your choice : ");
            scanf("%d",&ch);
            return ch;
    }
    void main()
    {
            int ch;
            node *newnode;
            do
            {
                    ch=menu();
                    switch(ch)
                    {
                            case 1 :  newnode=getnode();
                                      push(newnode);
                                      break;
                            case 2 :  pop();
                                      break;
                            case 3 :  display();
                                      break;
                            case 4 :  return;
                    }
                    getch();
            }while(1);
    }
```

17.6 INPUT/OUTPUT:

```
STACK OPERATIONS USING POINTERS
 -------------******-------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 1
 enter data : 10
                              data pushed into Stack
STACK OPERATIONS USING POINTERS
 -------------******-------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 1
 enter data : 20
                                 data pushed into Stack

STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 1
 enter data : 30
                                 data pushed into Stack

STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 3
                                 elements in the stack..
           10   20   30

STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 2
                                 popped element is 30

STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 3
                                 elements in the stack..
           10   20


STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 2
                                 popped element is 20

STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 3
                                 elements in the stack..
           10
```

```
STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 2
                            popped element is 10

STACK OPERATIONS USING POINTERS
 --------------******--------------------
 1. Push
 2. Pop
 3. Display
 4. Quit
 enter your choice : 3
                stack empty
```

WEEK-18

18.1 OBJECTIVE:

Program to implement Queue

18.2 RESOURCES:

centOS

18.3 PROGRAM LOGIC:

1. **QUEUE:** Queue is a linear data structure which works under the principle of **first in first out. Basic operations: Insertion, deletion, display.**
2. **Inserion**: if (rear==MAX), display **Queue is full.** Else reading data and inserting at queue [rear], and doing rear++.
3. **Deletion:** if (front==rear), display **Queue is empty** .Else printing element at queue [front] and doing front++.
4. **Display:** if (front==rear), display **No elements in the queue** .Else printing the elements from queue[front] to queue[rear].

18.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

18.5 SOURCE CODE:

```
# include <stdio.h>
# include <conio.h>
# define MAX 4
int queue[MAX];
int front=0,rear=0;
int menu()
{
	int ch;
	printf("\n \t\t\t\t\tQueue using arrays:");
	printf("\n ****************************");
	printf("\n 1.Insertion");
	printf("\n 2.Deletion");
	printf("\n 3.Display");
	printf("\n 4.QUIT");
	printf("\n Enter your choice:");
	scanf("%d",&ch);
	return ch;
}
void insertq()
{
	int data;
	if(rear==MAX)
	{
		printf("\n Queue is full");
		return;
	}
	else
	{
		printf("\n Enter data:");
		scanf("%d",&data);
		queue[rear]=data;
		rear=rear+1;
```

```
                printf("\n Data entered successfully");
        }
}
void deleteq()
{
        if(front==rear)
        {
                printf("\n Queue is empty");
                return;
        }
        else
        {
                printf("\n The deleted element from the queue is:%d",queue[front]);
                front=front+1;
        }
}
void displayq()
{
        int i;
        if(front==rear)
        {
                printf("\n No elements in the queue");
                return;
        }
        else
        {
                printf("\n The elements in the queue are:");
                for(i=front;i<rear;i++)
                        printf("%5d",queue[i]);
        }
}
void main()
{
        int ch;
        do
        {
                ch=menu();
                switch(ch)
                {
                        case 1:insertq();
                                break;
                        case 2:deleteq();
                                break;
                        case 3:displayq();
                                break;
                        default:printf("\n U HAVE QUITED OUT");

                }
        }while(ch==1||ch==2||ch==3);
        getch();
}
```

18.6 INPUT/OUTPUT:

```
1.Insertion
2.Deletion
3.Display
```

```
4.QUIT
Enter your choice:1

Enter data:25

Data entered successfully
Queue using arrays:
*****************
1.Insertion
2.Deletion
3.Display
4.QUIT
Enter your choice:2

The deleted element from the queue is:25
Queue using arrays:
****************
1.Insertion
2.Deletion
3.Display
4.QUIT
Enter your choice:3

The elements in the queue are:   36   25
Queue using arrays:
****************
1.Insertion
2.Deletion
3.Display
4.QUIT
```

18.7 PRE-LAB VIVA QUESTIONS:

1. What is a stack?
2. Stack data structure uses which principle?
3. Stack belongs to which type of data structure?
4. What do you mean by stack underflow?
5. What do you mean by stack overflow?
6. What are the basic operations of a stack?
7. How to implement stack?

18.8 LAB ASSIGNMENT:

1. Compose a program to implement stack and its operations using linked list.
2. Write a program to implement stack and its operations using arrays.
3. Formulate a program to reverse a list of numbers using stack.
4. Write a program to find the factorial of a number using stack.
5. Develop a program to check a given expression is balanced or not using stack.

18.9 POST-LAB VIVA QUESTIONS:

1. What is the time complexity of PUSH operation?
2. What is the time complexity of POP operation?
3. What are the applications of stack?
4. How to remove an element from stack?
5. How to insert an element using a stack?
6. Is it possible to store any number of data elements in stack?
7. What are the demerits of stack?

WEEK - 19

19.1 OBJECTIVE:

Program implement the linear regression algorithm.

19.2 RESOURCES:

centOS

19.3 PROGRAM LOGIC:

```
1. Read n
2. sumx = 0
3. sumxsq = 0
4. sumy = 0
5. sumxy = 0
6. for i = 1 to n do
7. Read x, y
8. sumx = sumx + x
9. sumxsq = Sumxsq + x2
10. sumy = Sumy + y
11. sumxy = sumxy + x * y end for
12. denom = n * sumxsq – sumx * sumx
13. a0 = (sumy * sumxsq – sumx * sumxy) / denom
14. a1 = (n * sumxy - sumx * sumy) / denonm
15. Write a1, a0
16. Stop
```

19.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

19.5 SOURCE CODE:

```
#include<stdio.h>
#include<math.h>
main()
{
        int n,I;
        float sumx, sumxsq, sumy, sumxy, x, y, a0, a1, denom;
       printf(“enter the n value”);
       scanf(“%d”, &n);
        sumx = 0;
        sumsq = 0;
       sumy = 0;
       sumxy = 0;
               for(i = 0; i < n; i++)
               {
               scanf(“%f %f”, &x, &y);
```

```
                    sumx += x;
                    sumsq += pow(x, 2);
                    sumy += y;
                    sumxy += x * y;
                    }
                    denom = n * sumxsq – pow(sumx, 2);
                    a0 = (sumy * sumxsq – sumx * sumxy) / denom;
                     a1 = (n * sumxy – sumx * sumy) / denom;
                     printf("y = %fx + %f",a1, a0);
}
```

19.6 INPUT/OUTPUT:

```
enter the n value 7
1 2
2 5
4 7
5 10
6 12
8 15
9 19
Y = 1.980769x + 0.096154
```

WEEK - 20

20.1 OBJECTIVE:

Program to implement the polynomial regression

20.2 RESOURCES:

centOS

20.3 PROGRAM LOGIC:

```
 1:  Start
2:  Read n
 3:  Initialize sumx = 0, sumxsq = 0, sumy = 0, sumxy = 0, sumx3 = 0, sumx4 = 0, sumxsq = 0
 4:  Initialize i = 0
5:  Repeat steps 5 to 7 until i < n
 6:  Read x, y
 7:  Sumx = sumx + x
         Sumxsq = sumxsq + pow(x, 2)
         Sumx3 = sumx3 + pow(x, 3)
         Sumx4 = sumx4 + pow(x, 4)
         Sumy = sumy + y
         Sumxy = Sumxy + x * y
         Sumxsqy = Sumxsqy + pow(x, 2) * y
 8:  Increment I by 1
9:  Assign
         a[0][0] = n
         a[0][1] = n
         a[0][2] = n
         a[0][3] = n
         a[1][0] = n
         a[1][1] = n
         a[1][2] = n
         a[1][3] = n
         a[2][0] = n
         a[2][1] = n
         a[2][2] = n
         a[2][3] = n
10: Initialize i = 0
11: Repeat steps 11 to 15 until i < 3
 12: Initialize j = 0
 13: Repeat step 13 to 14 until j <= 3
 14: Write a[i][j]
 15: Increment j by 1
16: Increment I by 1
 17: Initialize k = 0
18: Repeat steps 18 to 27 until k <= 2
19: Initialize i = 0
```

```
 20: Repeat step 20 to 26 until i <= 2
21: If I not equal to k
22: Assign u = a[i][k] / a[k][k]
23: Initialize j = k
24: Repeat steps 24 and 25 until j <= 3
 25: Assign a[i][j] = a[i][j] – u * a[k][j]
26: Increment j by 1
27: Increment i by 1
28: Increment k by 1
29: Initialize I = 0
30: Repeat steps 31 to 33 until i < 3
31: Assign b[i] = a[i][3] / a[i][i]
32: Write I, b[i]
33: Increment I by 1
34: Write b[2], b[i], b[0]
 35: Stop
```

20.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

20.5 SOURCE CODE:

```
#include<stdio.h>
#include<math.h>
main()
{
        int n, I, j, k;
       float sumx, sumxsq, sumy, sumxy, x, y;
        float sumx3, sumx4, sumxsqy, a[20][20], u = 0.0, b[20];
        printf("\n Enter the n value");
        scanf("%d", &n);
        sumx = 0;
        sumxsq = 0;
        sumy = 0;
        sumxy = 0;
        sumx3 = 0;
       sumx4 = 0;
       sumxsqy = 0;
        for(i = 0; i < n; i++)
        {
        scanf("%f %f", &x, &y);
       sumx += x;
       sumxsq += pow(x, 2);
```

```
        sumx3 += pow(x, 3);
         sumx4 += pow(x, 4);
          sumy += y;
         sumxy += x * y;
         sumxsqy += pow(x, 2) * y;
    }
         a[0][0] = n;
         a[0][1] = sumx;
         a[0][2] = sumxsq;
        a[0][3] = sumy;
        a[1][0] = sumx;
         a[1][1] = sumxsq;
        a[1][2] = sumx3;
        a[1][3] = sumxy;
        a[2][0] = sumxsq;
         a[2][1] = sumx3;
         a[2][2] = sumx4;
        a[2][3] = sumxsqy;
        for(i = 0; i < 3; i++)
        {
         for(j = 0; j <= 3; j++)
         printf("%10.2f", a[i][j]);
        printf("\n");
         }
        for(k = 0; k <= 2; k++)
        {
            for(i = 0; i <= 2; i++)
            {
            if(i != k)
             u = a[i][k]/a[k][k];
            for(j = k; j <= 3; j++)
            a[i][j] = a[i][j] – u * a[k][j];
             }
        }
             for(i = 0; i < 3; i++)
        {
             b[i] = a[i][3]/a[i][i];
            printf("\nx[%d] = %f", I, b[i]);
        }
            printf("\n");
            printf("y = %10.4fx + 10.4 fx + %10.4f", b[2], b[1], b[0]);
        }
```

20.6 INPUT/OUTPUT:

```
Enter the n value 10
-4 21
-3 12
-2 4
-1 1
0 2
1 7
2 15
3 30
4 45
5 67
10.00 5.00 85.00 204.00
5.00 85.00 125.00 513.00
85.00 125.00 1333.00 3193.00
X[0] = 2.030303
X[1] = 2.996970
X[2] = 1.984848
Y = 1.9848xsq + 2.9979x + 2.0303
```

WEEK -21

21.1 OBJECTIVE:

Program to implement the Lagrange interpolation**.**

21.2 RESOURCES:

centOS

21.3 PROGRAM LOGIC:

21.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

21.5 SOURCE CODE:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
        void main()
        {
                float x[10],y[10],temp=1,f[10],sum,p;
                int i,n,j,k=0,c;
                clrscr();
               printf("\nhow many record you will be enter: ");
               scanf("%d",&n);
               for(i=0; i<n; i++)
               {
                printf("\n\nenter the value of x%d: ",i);
                scanf("%f",&x[i]);
                printf("\n\nenter the value of f(x%d): ",i);
                scanf("%f",&y[i]);
                }
               printf("\n\nEnter X for finding f(x): ");
               scanf("%f",&p);
                for(i=0;i<n;i++)
                {
                        temp = 1;
                        k = i;
                 for(j=0;j<n;j++)
                        {
                                if(k==j)
                                {
                                continue;
                                }
                else
```

```
                        {
                                    temp = temp * ((p-x[j])/(x[k]-x[j]));
                        }
            }
                                   f[i]=y[i]*temp;
            }
                         for(i=0;i<n;i++)
                         {
                         Sum = sum + f[i];
                        }
                         printf("\n\n f(%.1f) = %f ",p,sum);
                         getch();
}
```

21.6 INPUT/OUTPUT:

```
how many record you will be enter: 4
enter the value of x0: 0
enter the value of f(x0): 0
enter the value of x1: 1
enter the value of f(x1): 2
enter the value of x2: 2
enter the value of f(x2): 8
enter the value of x3: 3
enter the value of f(x3): 27
Enter X for finding f(x): 2.5
 f(2.5) = 15.312500
```

WEEK - 22

22.1 OBJECTIVE:

To implement the Newton- Gregory forward interpolation.

22.2 RESOURCES:

centOS

22.3 PROGRAM LOGIC:

```
1.  Start
2.  Read n
3.  for i = 0 to (n-1) do read xi, yi
4.  read x
5.  h ← xi-x0
6. p ← (x - xo)/n
7 .  for j = 0 to n-2 do
       Δ1yj ← yj + 1 - Δi - 1
8.  k ← n - 2
9.  for i = 2 to (n - 1) do
     9.1:  k ← k - 1
     9.2:  for j = 0 to k do
       Δiyj ← Δi - 1 yj + 1 - Δi - 1yj
10.  Sumy ← y0
11.  Pvalue ← 1
12.  Fact value ← 1
13.  for l = 1 to (n - 1) do
        13.1: Pvalue ← pvalue x (p - (l - 1))
        13.2: factvalue ← factvaluex1
        13.3: term ← (pvalue x Δly) / factvalue
        13.4: Sumy ← Sumy + term
14:  Print x, SUMY
15:  Stop
```

22.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

22.5 SOURCE CODE:

```
#include<stdio.h>
#include<math.h>
main()
{
        int i, j, n, k, l;
        float sumy, h, term, p, z, pvalue;
       float x[25], y[25], d[25][25], factvalue;
        printf("enter the value of n");
        scanf(“%d”, &n);
        printf(“enter %d values for x, y \n”, n);
       for(i = 0; i < n; i++)
        scanf(“%f %f”, &x[i], &y[i]);
        printf(“\n enter z”);
        scanf(“%f”, &z);
        h = x[1] – x[0];
        p = (z - x[0] )/ h;
        for(j = 0; j < n-2; j++)
                d[i][j] = y[j + 1] – y[j];
        k = n-2;
        for(i = 2; i < n; i++)
        {
                k++;
               for(j = 0; j <= k; j++)
                 d[i][j] = d[i - 1][j + 1] – d[i - 1][j];
        }
        for(l = 1; l < n; l++)
        {
               pvalue *= (p - (l - 1));
               factvalue *= 1;
               term = pvalue * d[l][0] / factvalue;
               sumy += term;
        }
        printf(“\n y value at z = %f is %f”, z, sumy);
}
```

Input & Output:

```
enter n 7
enter 7 data values for x, y
1921 35
1931 42
1941 58
1951 84
1961 120
1971 165
1981 220
enter z 1925
y value at z = 1925.000000 is 36.756710
```

22.6 INPUT/OUTPUT:

```
enter n 7
enter 7 data values for x, y
1921 35
```

```
1931 42
1941 58
1951 84
1961 120
1971 165
1981 220
enter z 1925
y value at z = 1925.000000 is 36.756710
```

WEEK - 23

23.1 OBJECTIVE:

To implement Trapezoidal method.

23.2 RESOURCES:

centOS

23.3 PROGRAM LOGIC:

```
1.  Read x1, x2, e {x1 and x2 are the two end points of the internal the allowed error in integral is e}
2.  h = x2 - x1
3.  SI = (f(x1) + f(x2))/2;
4.  I = h - si
5.  i = 1 Repeat
6.  x = x1 + h/2
7.  for J= 1 to I do
8.  SI = SI + f(x)
9.  x = x + h
    End for
10. i = 21
11. h = h/2 {Note that the internal has been halved above and the number of points where the function has to be computed is doubled}
12. i0 = i1
13. i1 = h.si
14. until / I1 - i0 / <= c./i1/
15. Write I1, h, i
16. Stop.
```

24.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

24.5 SOURCE CODE:

```
#include<stdio.h>
#include<math.h>
main()
{
        float h, a, b, n, x[20], y[20], sum = 0, integral;
        int i;
        clrscr();
        printf("enter the value of a, b, n:");
        scanf("%f %f %f", &a, &b, &n);
        printf("enter the values of x:");
        for(i = 0; i <= (n-1); i++)
        {
                scanf("%f", &x[i]);
```

```
            }
             printf("\n enter the values of y:");
             for(i = 0; i <= (n-1); i++)
            {
                        scanf("%f", &y[i]);
             }
            h = (b-a)/n;
             x[0] = a;
             for(i = 1; i <= n-1; i++)
            {
                          x[i] = x[i-1] + h;
                        sum = sum + 2 * y[i];
             }
             sum = sum + y[b];
             integral = sum * (h/2);
             printf("approximate integral value is: %f", integral);
             getch();
}
```

25.6 INPUT/OUTPUT:

```
enter the values of a, b, n
123
enter the values of x:
123
enter the values of y:
123
approximate integral value is 2.166667
```

WEEK- 24

24.1 OBJECTIVE:

To implement Simpson method**.**

24.2 RESOURCES:

centOS

24.3 PROGRAM LOGIC:

```
1. Read x1, x2, e
2. h = (x2 - x1)/2
3. i = 2
4. si = f(x1) + f(x2)
5. s2 = 0
6. s4 = f(x1 + h)
7. I0 = 0
8. In = (s + 4s4).(h/3)
Repeat
9. s2 = s2 + s4 { s2 stores already computed functional value and s4 the value computed in the new nitration }
10. s4 = 0
11. x = x1 + h/2
12. for j = 1 to I do
13. s4 = s4 + f(x)
14. x = x + h
15. h = h/2
16. i = 2i
17. io = in
18. in = (s1 + 2s2 + 4s4) . (h/3)
19. until |In-Io|≤e. /in
20. Write In, h, i
21. STOP
```

24.4 PROCEDURE:

1. Create : Open editor vi x.c write a program after that press ESC and :wq for save and Quit.
2. Compile: gcc x.c.
3. Execute: ./ a.out

24.5 SOURCE CODE:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
main()
{
        float h, a, b, n, x[20], y[20], sum = 0, itgl;
         int i;
         clrscr();
        printf("enter the values of a, b, n");
```

```
            scanf("%f%f%f", &a, &b, &n);
             printf("enter the values of x");
             for(i = 0; i <= n; i++)
            {
                        scanf("%f", &x[i]);
             }
             printf("\n enter the values of y");
             for(i = 0; i <= n; i++)
             {
                        scanf("%f", &y[i]);
             }
             h = (b - a)/n;
             a = x[0];
             b = x[n];
             for(i = 0; i <= (n-2); i++)
            {
                         x[i] = x[i] + h;
                         if(i % 2 == 0)
                        {
                                    sum = sum + 4 * y[i];
                         }
                        else
                         {
                                    sum = sum + 2 * y[i];
                          }
             }
             itgl = sum * (h/3);
             printf("integral value%f", itgl);
            getch();
}
```

24.6 INPUT/OUTPUT:

```
enter the values of a, b, n
123
enter the value of x
4567
enter the values of y
8912
integral value is 5.555556
```

VIVA QUESTIONS

1) Who invented C Language ?

Ans: Dennis Ritchie in 1972 developed a new language by inheriting the features of both BCPL and B and adding additional features. He named the language as just C

2) Who invented B Language ?

Ans: Ken Thomson at AT&T Bell Labs developed a language and named it B. Even the B language was found to have some short comings to support development of both business applications and system software.

3) Who invented BCPL Language ?

Ans: Basic Combined Programming Language(BCPL) was developed by Martin Richards, Cambridge University.

4) Why C Language ?

Ans: C is one of the high level languages. It is a general purpose language, which means it can be used to write programs of any sort.

5) What are the advantages of c language ?

Ans: The advantages of c language are as follows:

- Easy to write
- Rich set of operators and functions that are built–in
- Support for bit–wise operation
- Flexible use of pointers
- Direct control over the hardware
- Ability to access BIOS/DOS routines
- Interacting using Interrupts
- Ability to create library files (.LIB)
- Ability to write interface programs
- Incorporating assembly language in C program

6) What are the disadvantages of c language ?

Ans: The disadvantages of C language are as follows:

- C is considered difficult to learn
- Because of its conciseness, the code can be difficult to follow
- It is not suited to applications that require a lot of report formatting and data file manipulation

7) What are the salient features of c languages ?

Ans: The following are the salient features of C language :

- C is called a middle level language
- C supports structured design approach
- C is extensible
- C is rich in data types and operators
- C is portable

8) What is a header file ?

Ans: Header files provide the definitions and declarations for the library functions. Thus, each header file contains the library functions along with the necessary definitions and declarations. For example, stdio.h, math.h, stdlib.h, string.h etc.

9) What is character set ?

Ans: Character set is the set of characters allowed and supported in the programming language. Generally a program is a collection of instructions, which contain groups of characters. Only a limited set of characters is allowed to write instructions in the program.

10) What is C token ?
Ans: The smallest individual units of a C program are known as tokens.

11) List the different types of C tokens ?
Ans: The different types of C tokens are:
- Constants
- Identifiers
- Keywords
- Operators
- Special symbols
- Strings

12) What is a string ?
Ans: A string is a sequence of characters ending with NUL. It can be treated as a one–dimensional array of characters terminated by a NUL character.

13) What are qualifiers ?
Ans: Qualifiers or modifiers are identifiers that may precede the scalar data types (except float) to specify the number of bits used for representing the respective type of data in memory. The qualifiers in C are short, long, signed, and unsigned.

14) What is a function ?
Ans: A function is a set of statements to perform a specific task.

15) What is a constant ?
Ans: A constant is a value that does not change during the program execution. A constant used in C does not occupy memory.

16) What is variable ?
Ans: An identifier is used to identify and store some value. If the value of the identifier is changed during the execution of the program, then the identifier is known as variable.

17) What are the rules for the identifier ?
Ans: Naming rules for an identifier are:
- The first character must be an alphabet or underscore (_)
- Digits may be included in the variable
- The maximum number of characters in a word are 32 (It may vary depending upon the platform)
- No other special characters are allowed.

18) What are global variables ?
Ans: Global Variables are those, which are required to be acccessed by all the functions defined after their declaration. So, the variables declared before the main {} can be acccessed by all the functions, which follow their declaration.

19) What is a keyword ?
Ans: Keywords are those words of C which have predefined meaning assigned by the C language. They form a part of the database required by the C compiler.

20) What are the different types of c instructions ?
Ans: There are basically three types of instructions in C are :
- Type Declaration Instruction
- Arithmetic Instruction
- Control Instruction

21) What is an expression ?

Ans: Expression is defined as a combination of operands and operators to obtain some computation. Operands represent variables or values and The operator tells is what operation to be performed.

22) What are the types of data files ?

Ans: There are two types of data files :

- stream oriented or standard data files
- system oriented or low level data files

23) Why C is called a middle level language ?

Ans: C combines the features of both Assembly Level Languages (Low Level Languages) and Higher Level Languages. For this reason, C is referred to as a Middle Level Language. The feature of ALLs is that of enabling us to develop system level programs and the features of HLLs are those of higher degree of readability and machine independence.

24) How can variables be characterized ?

Ans: The variables can be categorized by storage class as well as by data type. The storage class specifies the portion of the program within which the variables are recognized.

25) What is the purpose of type declarations ?

Ans: The type declaration allow to create a synonym for other data types. Its syntax is typedef type identifier; The declaration typedef unsigned long int INTEGER

26) What is recursion ?

Ans: C language a function may call another function. When a function calls itself, it is referred to as recursive call and the process is known as recursion. C provides very good facilities for recursion.

27) What is data types ?

Ans: Data types refer to the classes of data that can be manipulated by C programs. The three fundamental data types supported by C are character, integer and real type.

28) What are the types of macro formats ?

Ans: There are two types of macro formats. There are

- Simple
- Parameterized

29) What are the different types of errors ?

Ans: The different types of errors are:

- Compile–Time Errors
- Linker Errors
- Runtime Errors
- Logical Errors

30) What is meant by errors and debugging ?

Ans: Errors may be made during program creation even by experienced programmers. Such type of errors are detected by the compiler. Debugging means removing the errors..

31) What is the purpose of main() function ?

Ans: The function main() invokes other functions within it.It is the first function to be called when the program starts execution.

- It is the starting function.
- It returns an int value to the environment that called the program.
- Recursive call is allowed for main( ) also.
- It is a user-defined function.

32) What is meant by type casting ?
Ans: It is the explicit type conversion required for a number before carrying out processing or assigning to another variable.

33) What are the primitive data types in c ?
Ans: There are five different kinds of data types in C.

- Char
- Int
- Float
- Double
- Void

34) What is the use of typedef ?
Ans: The typedef help in easier modification when the programs are ported to another machine. A descriptive new name given to the existing data type may be easier to understand the code.

35) What is meant by type specifiers ?
Ans: Type specifiers decide the amount of memory space occupied by a variable. In the ease of integral types; it also explicitly states the range of values that the object can hold..

36) What is masking ?
Ans: Masking is a process in which a given bit pattern is partly extracted into another bit pattern by means of a logical bitwise operation.

37) What is the difference between single character constant and string constant ?
Ans: A single character constant consists of only one character and it is enclosed within a pair of single quotes. A string constant consists of one or more characters and it is enclosed within a pair of double quotes.

38) What is signed and unsigned ?
Ans: A numeric value, may have a positive or a negative sign. In the memory, for a variable, one bit is used exclusively to maintain the sign of the data. If we don't have sign, the sign bit also may be used for data. If the value is negative, the sign bit is 1, and if it is positive, it will be 0.

39) What is this pointer ?
Ans: It is a pointer that points to the current object. This can be used to access the members of the current object with the help of the arrow operator

40) What is zero based addressing ?
Ans: The array subscripts always start at zero. The compiler makes use of subscript values to identify the elements in the array. Since subscripts start at 0, it is said that array uses zero-based addressing.

41) What is a loop ?
Ans: A loop is a process to do a job repeatedly with possibly different data each time. The statements executed each time constitute the loop body, and each pass is called iteration. A condition must be present to terminate the loop.

42) What is break statement ?
Ans: When a break is encountered inside a loop, the loop is terminated and the control passes to the statement following the body of the loop.

43) What is the use of getchar() function ?
Ans: It returns a character just entered from the standard input unit, that is, keyboard. The entered character can be either assigned to a character variable or echoed to the computer screen.

44) How to declare pointer variables ?

Ans: If a variable is going to be a pointer, it must be declared as such. A pointer declaration consists of a base type, an *, and the variable name. The general form for declaring a pointer variable is data _type * var_ name;

45) What is the difference between fread buffer() and fwrite buffer() ?
Ans: Fread(), buffer is a pointer to an area of memory that will receive the data from the file. For fwrite(), buffer is a pointer to the information that will be written to the file. The value of count determines how many items are read or written, with each item being num_byte bytes in length. The size_t in both the formats is defined as some kind of unsigned integer. Finally, fp is a file pointer to a previously opened file.

46) What is macro ?
Ans: The second preprocessor function is macro definition. A macro is formal syntax that can be used to generate statements for use in a program. For the C language, the macro generates C statements.

47) What are the types of I/O functions ?
Ans: I/O functions are grouped into two categories :

- Unformatted I/O functions
- Formatted I/O functions

48) What is the difference b/w formatted & unformatted I/O functions ?
Ans: The formatted I/O functions allow programmers to specify the type of data and the way in which it should be read in or written out. On the other hand, unformatted I/O functions do not specify the type of data and the way is should be read or written.

49) How to declare pointer variables ?
Ans: If a variable is going to be a pointer, it must be declared as such. A pointer declaration consists of a base type, an *, and the variable name. The general form for declaring a pointer variable is data _type * var_ name;

50) What is the use of putchar function ?
Ans: The putchar function displays one character on the display monitor. The character to be displayed is of type char. The syntax for putchar function is as given below :putchar (ch_var);Where ch_var is a previously declared character variable.

51) What is the use of getchar functions ?
Ans: The getchar function accepts a single character from the keyboard. The function does not require any arguments, though a pair of empty parentheses must follow the word getchar as a syntax. It returns a single character from a standard input device (typically a keyboard) and it can be assigned to predeclared character variable.

52) What is character constants ?
Ans: A character constant is a single character, enclosed within the pair of single quotation mark (apostrophes).

53) What is string constants ?
Ans: A string constant or literal contains a sequence of zero or more characters or escape seauences enclosed in double Quotation marks.
54) What is integer constants ?
Ans: An integer constant is an integer-valued number. It can represent decimal, octal, or hexadecimal values.

55) What is the difference between fread and fwrite function ?
Ans: The fread() function returns the number of items read. This value may be less than count if the end of the file is reached or an error occurs. The fwrite() function returns the number of items written. This value will equal count unless an error occurs.

56) What are linker error ?

Ans: The Linker Errors occur during the linking process when the external symbols referred to by the program are not resolved.

57) What are runtime error ?
Ans : The Runtime Errore occur while a program is being run and hence the name. they occur due to both program internal and external factors.

58) When do we get logical errors ?
Ans: The Logical Errors occur if the solution procedure for the given problem itself is wrong.

- In this case, the outputs produced by the programs would be incorrect.
- Correcting the solution procedure itself by better understanding of the problem eliminates these errors.
- The Logical Errors (if any) are to be figured out by ourselves by verifying the outputs that are produced by the program.

59) Do character constants represent numerical values ?
Ans: Yes, each character constant associates an integer value with it.

60) What is the purpose of scanf() and printf() functions ?
Ans: The function scanf() is used for formatted input from the standard input and provides many of the conversion facilities.It is used for formatted output to standard output device, that is, screen. The format specification string and the data to be output, are the arguments (parameters) to the printf() function.

61) What is type qualifiers ?
Ans: Type qualifier adds properties to an identifier. Type qualifiers describe the manner in which the object will be modified. The application of qualifiers to an object does not affect the range or the arithmetic properties of the object.

62) What are the types of type qualifiers in c ?
Ans: The two type qualifiers provided by C are :

- const
- volatile

63) What is meant by operator precedence ?
Ans: Operator precedence describes the order in which C evaluates different operators in a complex expression.

64) What is an Operator ?
Ans: An operator is a symbol, which instructs the computer to perform the specified manipulation over some data. The rich set of operators available in C enable us to write efficient and concise programs and this fact serves to set C apart from any other programming languages.

65) What is a ternary operator in C ?
Ans: Perhaps the most unusual operator in C language is one called the conditional expression operator. Unlike all other operators in C which are either unary or binary operators the conditional expression operator is a ternary operator; that is, it takes three operands. The two symbols that are used to denote this operator are the question mark (?) and the colon (:). The first operand is placed before the ?, the second between the ? and the and the third after the :.

66) What is assignment operator ?
Ans: An operator is a symbol that operates on a certain data type.In C, the ’=’ symbol is known as the assignment operator. It sots the value of the variable on the left hand side of it to that of the right hand side of it.

67) What are the types of assignment statements ?
Ans: C supports a variety of assignment statements. These are given below :

- Simple assignment statement
- Multiple assignment statement

- Arithmetic assignment statement

68) What is the sizeof () operator ?
Ans: Even though it looks like a keyword, sizeof( ) is an operator which is used to know the memory size of the data types and variables. This operator returns the number of bytes allocated for the variable (or) data type.

69) What is the use of bitwise operator ?
Ans: The bitwise operator performs the operation on bits (i.e bit by bit). Using the bitwise operators we can set/reset/check any bit in the value of the variable.

70) What is the Difference between = and = = Operators ?
Ans: The two operators = and == are used for assignment and checking respectively. If not properly used, it causes many problems. The following program illustrates what will happen if we use = instead of = =.

71) What is unary operator ?
Ans: The operators that act upon a single operand to produce a new value are known as unary operators.

72) What is the difference between break and continue ?
Ans: The break statement is used to exit from all the loop constructs (while, do while and for) and switch.case statements, whereas the continue statement is used to skip all subsequent instructions and can control back to the loop control. The continue statement can be used for any loop construct.

73)What is storage class ?
Ans: The storage class in C provides the complete information about the location and visibility of variables. Scope of a variable means the portion of the program within which it can be referenced and lifetime means the time of its existence in the memory.

74) What are the different storage classes in C ?
Ans: There are four types of storage classes.

- Automatic : Variable used as a local variable. This is the default one. Initial value of variable is garbage value without initialization.
- Extern : Variable used as a local variable. Retains its value during next function call.
- Regiter : Variable used as a local variable. May be stored in register if possible. Default initial value is garbage value.
- Static : Variable used as a global variable.

75) What is getche() function ?
Ans: It returns a character just entered from the standard input unit. The entered character is echoed (displayed) to the computer screen. It reads a single character the moment it is typed without waiting for the Enter key to be hit.

76) What is getch() function ?
Ans: It returns a character just entered from the standard input unit. The entered character is not echoed on the screen. It reads a single character the moment it is typed without waiting for the Enter key to be hit.

77) What is an arrays ?
Ans: Arrays can be defined as a collection of variables of the same type that are referred through a common name.

78) What are the advantages of the functions ?
Ans: The advantages using functions are:

- It reduces the complexity in a program by reducing the code
- Function are easily understanding and reliability and execution is faster
- It also reduces the time to run a program. In other way, Its directly proportional to complexity
- Its easy to find-out the errors due to the blocks made as function definition outside the main function.

79) What is the difference between arrays and pointers ?
Ans: Array is collection of similar data type. It is a static memory allocation means we cannot increment and decrement the array size once we allocated. And we cannot increment the base address, reassign address. Pointer is a dynamic memory allocation. we can allocate the size as we want, assigning into another variable and base address incrimination is allowed.

80) What is else if ladder ?
Ans: The else if ladder helps select one out of many alternative blocks of statements for execution depending on the mutually exclusive conditions.