CONTENTS

BIGGEST OF THREE NUMBERS

Expt.No-1 a) **Date: 05-07-2021**

AIM:

To Find out the biggest of given three numbers using else if ladder.

PROCEDURE:

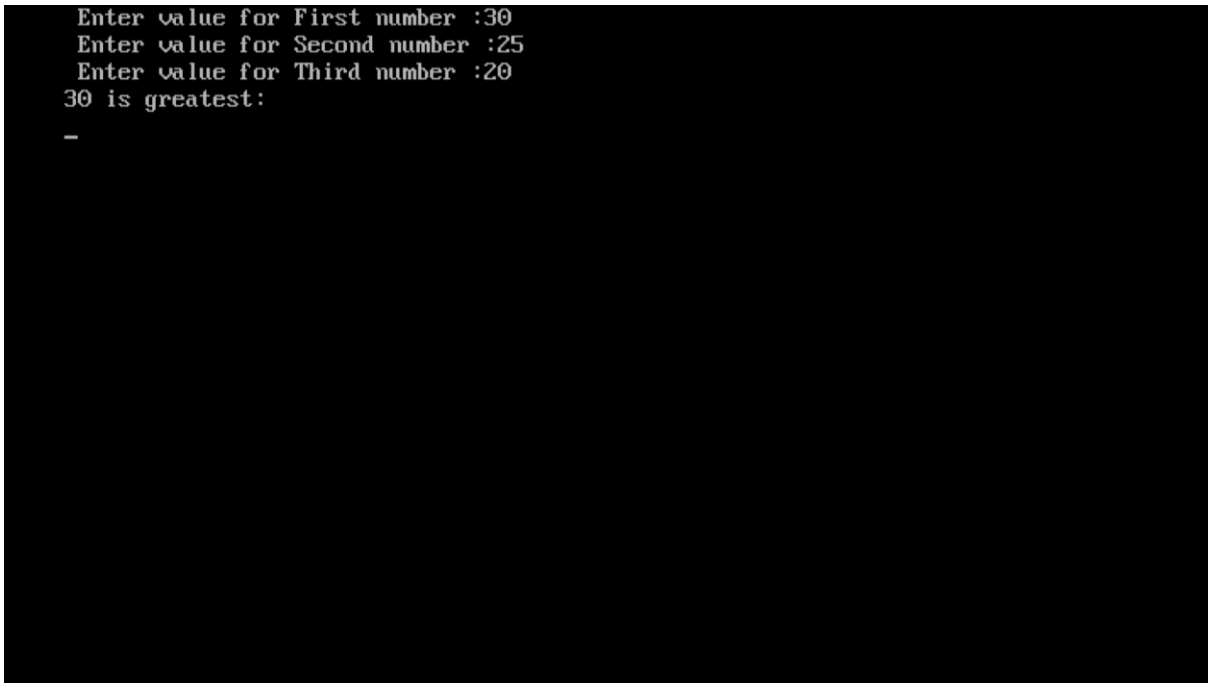
Step-1	**Start** the Program
Step-2	**Declare** num1, num2, num3 as integer
Step-3	**Read** num1,num2 and num3 values
Step-4	**If** num1 is greater than num2 and num3 then
Step-5	**Display** num1 is bigger and **goto** Step-9
Step-6	**Else** if num2 is greater than num1 and num3 then
Step-7	**Display** num2 is bigger and **goto** Step-9
Step-8	**Else** Display num3 is bigger.
Step-9	**END of the Program**

PROGRAM

```
#include<conio.h>
#include<iostream.h>
void main()
{
        int num1,num2,num3;
clrscr();
        cout<<" Enter value for First number :";
        cin>>num1;
        cout<<" Enter value for Second number :";
        cin>>num2;
        cout<<" Enter value for Third number :";
        cin>>num3;
        if((num1>num2)&& (num1> num3)) {
```

```
                cout<<num1<<" is greatest:"<<endl;
        } else if((num2>num1) && (num2>num3)) {
                cout<<num2<<" is greatest:"<<endl;
        } else {
                cout<<num3<<" is greatest:"<<endl;
        }
getch();
   }
```

SAMPLE I/O



RESULT:

The output has been verified successfully.

DELETING VOWELS IN A GIVEN STRING

Expt.No-1 b) **Date: 05-07-2021**

AIM:

To Delete the vowels in a given string using Control Statements.

PROCEDURE:

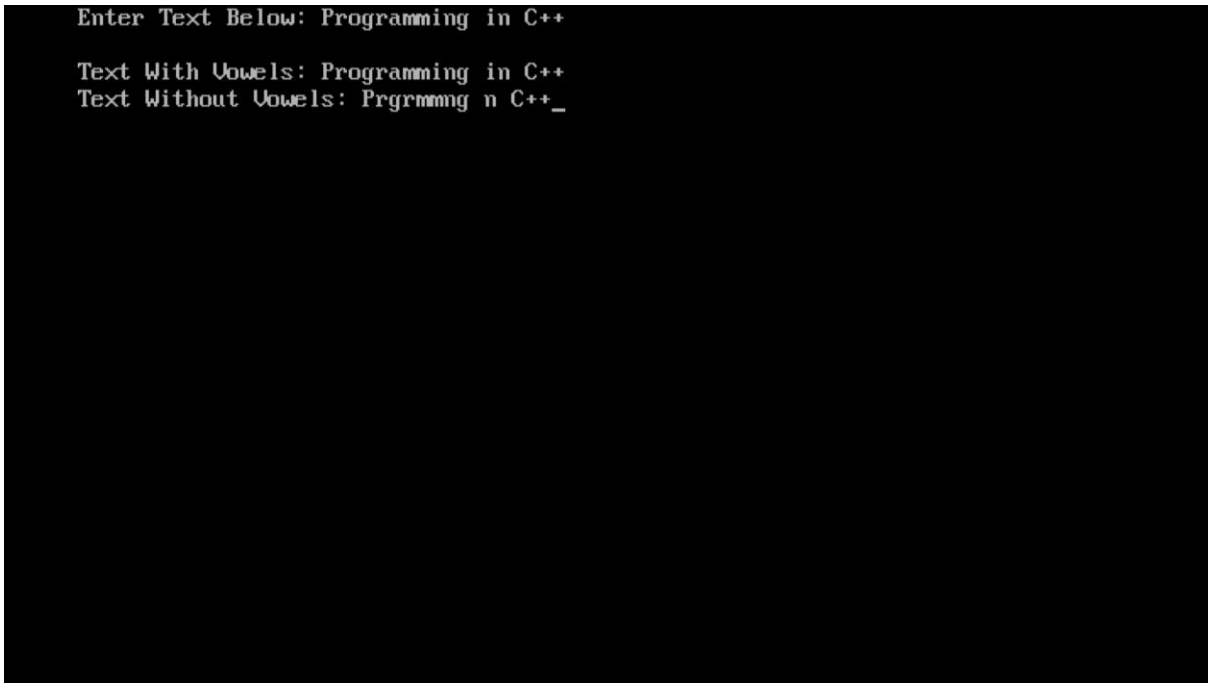
Step-1	**Start** the Program
Step-2	**Declare** str_in and str_out as string
Step-3	Declare i, j, len as integer
Step-4	**Read** str_in value
Step-5	**Calculate** the length of given string using **for** loop
Step-6	**Initialize** i=0 and j=0
Step-7	**Repeat** steps to until i<length
Step-8	**If** whether the str[i] is vowels then **do Step-7**
Step-9	**Else**, Assign str_out[j]= str_in **do Step-7**
Step-10	**Display** the str_in, str_out values
Step-11	**END of the Program**

PROGRAM

```
#include<iostream.h>
#include<conio.h>
void main()
{
        char str_in[80],str_out[80];
        int i,j,len;
clrscr();
        cout<<" enter text below: ";
        cin.getline(str_in,25);
        for(len=0;str_in[len]!='\0';len++);

        for(i=0,j=0;i<len;i++)
        {
                if(str_in[i]=='a'||str_in[i]=='e'||
```

```
                        str_in[i]=='i'  || str_in[i]=='o' ||
                        str_in[i]=='u' || str_in[i]=='A' ||
                        str_in[i]=='E' || str_in[i]=='I' ||
                        str_in[i]=='O' || str_in[i]=='U')
                {
                        continue;
                }
                else
                {
                        str_out[j]=str_in[i];
                        j++;
                }
        }
        str_out[j]='\0';
        cout<<"\n text with vowels: "<<str_in;
        cout<<"\n text without vowels: "<<str_out;
getch();
}
```

SAMPLE I/O



RESULT:

The output has been verified successfully.

IMPLEMENTING CLASS, OBJECTS AND MEMBER FUNCTIONS

Expt.No-2 **Date: 19-07-2021**

AIM:

To implement Student details using class, objects and member functions.

PROCEDURE:

Step-1	**Start** the Program
Step-2	**Declare** the class namely Student_details
Step-3	**Declare** class members reg_no, tot as integer.
Step-4	**Declare** class members marks as integer array
Step-5	**Declare** class members stud_name and course as character array.
Step-6	**Declare** Input() and Output() as public member functions
Step-7	Get class member values through Input member Function
Step-8	Display the class member values through Output member function
Step-9	Create an objects to Student_details class
Step-10	Call input() and Output().
Step-11	**END of the Program**

PROGRAM

```
 #include<iostream.h>
#include<conio.h>
class Student_details
{
        int reg_no,mark[5],tot;
        char stud_name[25],course[25];
public:
        void Input();
        void Output();
};
void Student_details:: Input()
        {
                cout<<" Enter the Reg no:";
```

```
                cin>>reg_no;
                cout<<" Enter the Student Name:";
                cin>>stud_name;
                cout<<" Enter the course:";
                cin>>course;
                tot=0;
                for(int i=0;i<5;i++)
                {
                        cout<<" Enter the Subject Marks";
                        cin>>mark[i];
                        tot+=mark[i];
                }
        }
void Student_details:: Output()
{
        cout<<" Reg No: "<<reg_no<<endl;
        cout<<" Name : "<<stud_name<<endl;
        cout<<" Course: "<<course<<endl<<endl;
        for(int i=0;i<5;i++)
        cout<<" Marks: "<<mark[i]<<endl;
        cout<<" Total :"<<tot<<endl;
}

void main()
{
clrscr();
        Student_details sd,sd1;
        cout<<"\n First Object :";
        sd.Input();
        sd.Output();
getch();
clrscr();
        cout<<"\n Second Object:";
        sd1.Input();
        sd1.Output();
getch();
}
```

SAMPLE I/O

```
First Object :
Enter the Reg no:21312
Enter the Student Name:Suresh
Enter the course:BCA
Enter the Subject Marks15
Enter the Subject Marks25
Enter the Subject Marks35
Enter the Subject Marks45
Enter the Subject Marks55
Reg No: 21312
Name : Suresh
Course: BCA

Marks: 15
Marks: 25
Marks: 35
Marks: 45
Marks: 55
Total :175
```

```
Second Object:
Enter the Reg no:21313
Enter the Student Name:Naresh
Enter the course:MCA
Enter the Subject Marks65
Enter the Subject Marks75
Enter the Subject Marks85
Enter the Subject Marks95
Enter the Subject Marks55
Reg No: 21313
Name : Naresh
Course: MCA

Marks: 65
Marks: 75
Marks: 85
Marks: 95
Marks: 55
Total :375
```

RESULT:

The output has been verified successfully.

IMPLEMENTING INLINE FUNCTIONS

Expt.No-3 **Date: 21-07-2021**

AIM:

To implement arithmetic operations of two class members using Inline functions.

PROCEDURE:

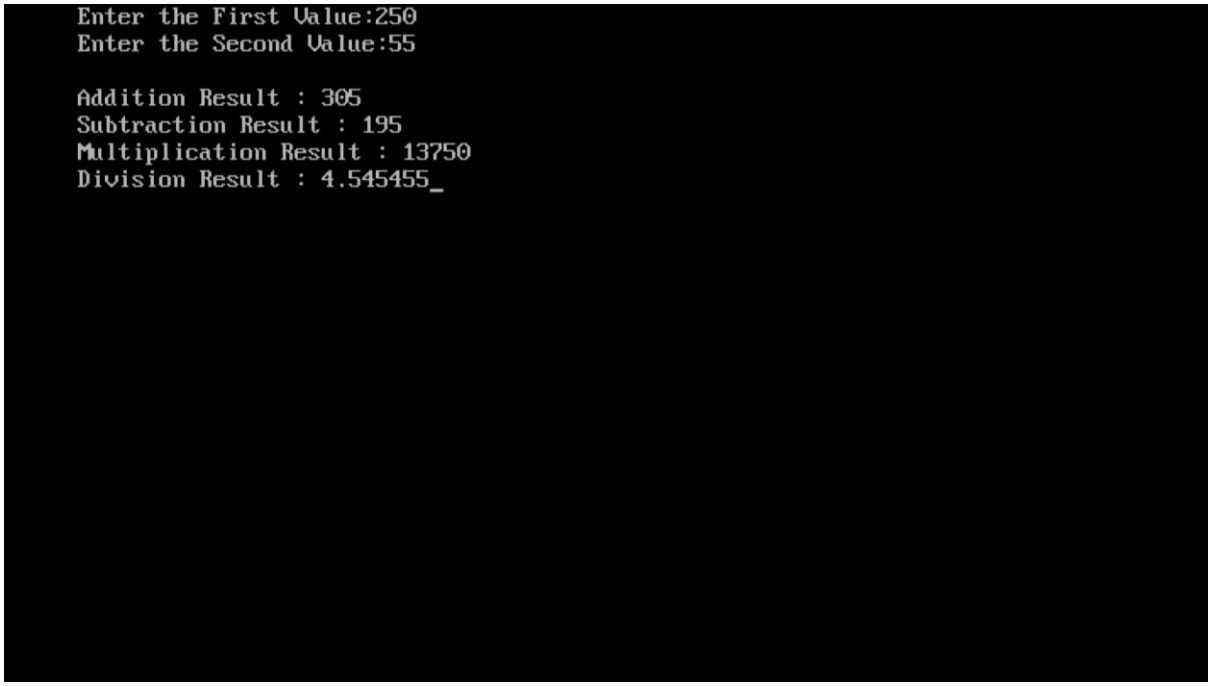
Step-1	**Start** the Program
Step-2	**Declare** the class namely numeric_values
Step-3	**Declare** class members value1,value2, result as integer.
Step-4	**Declare** Input() and Output() as public member functions
Step-5	**Get** class member values through Input member Function
Step-6	**Create** inline function namely Add(), Subtract(),Multiply(), Divide() .
Step-7	**Display** the class member values through Output member function
Step-8	**Create** an object to numeric_values class
Step-9	**Perform** the arithmetic operation based on the user input.
Step-10	**Call** input() and Output().
Step-11	**END of the Program**

PROGRAM

```
 #include<iostream.h>
#include<conio.h>
class numeric_values
{
        int value1,value2, result;
public:
        void Calc();
        void Output()
        {
                cout<<"\n Value 1: "<<value1<<endl;
                cout<<" Value 2: "<<value2<<endl;

        }
        inline void Add(int num1,int num2)
```

```
        {
                cout<<"\n Addition Result : "<<num1+num2;
        }
        inline void Sub(int num1,int num2)
        {
                cout<<"\n Subtraction Result : "<<num1-num2;
        }
        inline void Mul(int num1,int num2)
        {
                cout<<"\n Multiplication Result : "<<num1*num2;
        }
        inline void Div(int num1,int num2)
        {
                float result= float(num1)/ float(num2);
                cout<<"\n Division Result : "<<result;
        }
};
void numeric_values:: Calc()
{
        cout<<" Enter the First Value:";
        cin>>value1;
        cout<<" Enter the Second Value:";
        cin>>value2;
        Add(value1,value2);
        Sub(value1,value2);
        Mul(value1,value2);
        Div(value1,value2);

}
void main()
{
clrscr();
        numeric_values obj1;
        obj1.Calc();

getch();
}
```

SAMPLE I/O



RESULT:

The output has been verified successfully.

IMPLEMENTING CONSTRUCTORS AND DESTRUCTORS

Expt.No-4 **Date: 02-08-2021**

AIM:

To implement Constructors and Destructors to the class in C++ Program

PROCEDURE:

Step-1	**Start** the Program
Step-2	**Declare** the class Employee
Step-3	**Declare** the class members emp_no as long
Step-4	**Declare** the class members basic_pay, da, hra, pf,ta, tax,net_sal as float.
Step-5	**Declare** the class members emp_name and design as string
Step-6	**Create** an object obj2 for Employee class and initialize it using **Parameterized Constructor**.
Step-7	**Create** an object obj1 for Employee class and initialize it using **Default Constructor**.
Step-8	**Calculate** the net salary for Both objects
Step-9	**Display** the Net salary for both objects
Step-10	**Destructor** function called implicitly and release the memory
Step-11	**END of the Program**

PROGRAM

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
class Employee
{
private:
        long emp_no;
        float basic_pay,da,hra,pf,ta,tax,net_sal;
        char *emp_name,*design;
```

```
public:
        Employee()
        {
                cout<<"\n Default Constructor : ";
                cout<<" \n Enter the Employee Number : ";
                cin>>emp_no;
                emp_name = new char[20];
                cout<<" Enter the Employee Name : ";
                cin>>emp_name;
                design= new char[20];
                cout<<" Enter the Employee Designation: ";
                cin>>design;
                cout<<" Enter the Basic Pay: ";
                cin>>basic_pay;
                da=hra=pf=ta=tax=net_sal=0.0;
                Calc();
        }
        Employee(long Empno,char *Name,char *Design, float B_pay)
        {
                int length;
                emp_no=Empno;
                length= strlen(Name);
                emp_name= new char[length+1];
                strcpy(emp_name,Name);
                strcpy(design,Design);
                basic_pay= B_pay;
                da=hra=pf=ta=tax=net_sal=0.0;
                Calc();
        }
```

```
        void Calc()
        {
                da= (basic_pay*3)/100;
                hra=(basic_pay*8)/100;
                ta= (basic_pay*2)/100;
                pf= (basic_pay*1.5)/100;
                tax=(basic_pay*5)/100;
                net_sal= (basic_pay+da+hra+ta)- (pf+tax);
        }
        void Display()
        {
                cout<<"\n Employee Number : "<< emp_no;
                cout<<"\n Employee Name : "<<emp_name;
                cout<<"\n Employee Designation : "<<design;
                cout<<"\n Basic Salary : "<<basic_pay;
                cout<<"\n DA          : "<<da;
                cout<<"\n HRA          : "<<hra;
                cout<<"\n TA          : "<<ta;
                cout<<"\n PF          : "<<pf;
                cout<<"\n TAX          : "<<tax;
                cout<<"\n Net Salary   : "<<net_sal;
getch();
        }
        ~Employee()
        {
                cout<<"\n Destructor Called for "<<this->emp_no;
        }
};
```

```
void main()
{
clrscr();
        long eno;
        float bpay;
        char *name,*design;
        cout<<" Parameterized Constructor :";
        cout<<" \n Enter the Employee Number :";
        cin>>eno;
        cout<<" Enter the Employee Name :";
        cin>>name;
        cout<<" Enter the Employee Designation:";
        cin>>design;
        cout<<" Enter the Basic Pay :";
        cin>>bpay;
        Employee obj2(eno,name,design,bpay);
        Employee obj1;
        obj1.Display();
        obj2.Display();
getch();
}
```

OUTPUT

```
Parameterized Constructor :
Enter the Employee Number :21312
Enter the Employee Name :Anniebesant
Enter the Employee Designation:Doctor
Enter the Basic Pay :100000

Default Constructor :
Enter the Employee Number : 21313
Enter the Employee Name : Ponraj
Enter the Employee Designation: Scientist
Enter the Basic Pay: 90000

Employee Number : 21313
Employee Name : Ponraj
Employee Designation : Scientist
Basic Salary : 90000
DA           : 2700
HRA          : 7200
TA           : 1800
PF           : 1350
TAX          : 4500
Net Salary   : 95850
```

```
Employee Number : 21312
Employee Name : Anniebesant
Employee Designation : Doctor
Basic Salary : 100000
DA           : 3000
HRA          : 8000
TA           : 2000
PF           : 1500
TAX          : 5000
Net Salary   : 106500
```

RESULT

Thus, the program implements the Constructors and Destructors for Employee class.

IMPLEMENTING FUNCTION OVERLOADING

Expt.No-5 **Date: 03-08-2021**

AIM:

To implement function overloading for Arithmetic Class using C++ Program

PROCEDURE:

Step-1	**Start** the Program
Step-2	**Declare** the class Arithmetic
Step-3	**Declare** the class members num1,num2 as int
Step-4	**Declare** the class members first_str, second_str as string.
Step-5	**Declare** the class members value1, value2 as double.
Step-6	**Create** an object obj1 for Arithmetic class and initialize it using **Parameterized Constructor**.
Step-7	**Call Add ()** with two integer numbers and Display the Result
Step-8	**Call Add ()** with two string values and Display the Result
Step-9	**Call Add ()** with two double values and Display the Result
Step-10	**Call Add ()** with three integer numbers and Display the Result
Step-11	**Call Add()** with three string values and Display the Result
Step-12	**Call Add ()** with three double values and Display the Result
Step-13	**END of the Program**

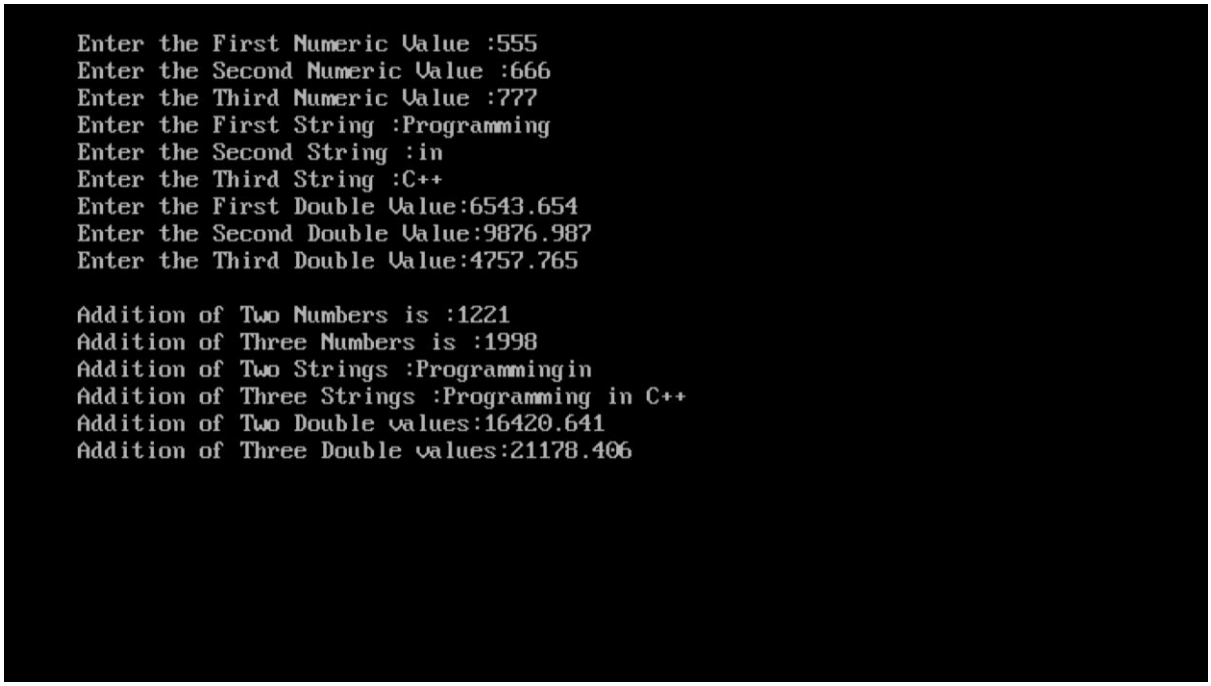
PROGRAM

```
#include<iostream.h>
#include<conio.h>
#include<string.h>

class Arithmetic
{
        int num1, num2,num3;
        char first_str[50],second_str[50],third_str[50];
        double value1,value2,value3;
public:
        Arithmetic(int num1,int num2,int num3,char first_str[50],
                char second_str[50],char third_str[50],double value1,
                double value2,double value3)
        {
        this->num1= num1;
        this->num2= num2;
        this->num3= num3;
        strcpy(this->first_str,first_str);
        strcpy(this->second_str,second_str);
        strcpy(this->third_str,third_str);
        this->value1= value1;
        this->value2= value2;
        this->value3= value3;
        }
        int Add(int value1, int value2)
        {
                return (value1+value2);
        }
```

```
        char* Add(char value1[50], char value2[50])
        {
                char res_str[150];
                strcpy(res_str,value1);
                strcat(res_str,value2);
                return(res_str);
        }
        double Add(double value1, double value2)
        {
                return(value1+value2);
        }
        int Add(int value1, int value2,int value3)
        {
                return (value1+value2+value3);
        }
        char* Add(char value1[50], char value2[50],char value3[50])
        {
                char res_str[150];
                strcpy(res_str,value1);
                strcat(res_str," ");
                strcat(res_str,value2);
                strcat(res_str," ");
                strcat(res_str,value3);
                return(res_str);
        }
        double Add(double value1, double value2,double value3)
        {
                return(value1+value2+value3);
        }
};
```

```
void main()
{
clrscr();
        int no1,no2,no3;
        char str[50],str1[50],str2[50];
        double val1,val2,val3;
        cout<<"\n Enter the First Numeric Value :";
        cin>>no1;
        cout<<" Enter the Second Numeric Value :";
        cin>>no2;
        cout<<" Enter the Third Numeric Value :";
        cin>>no3;
        cout<<" Enter the First String :";
        cin>>str;
        cout<<" Enter the Second String :";
        cin>>str1;
        cout<<" Enter the Third String :";
        cin>>str2;
        cout<<" Enter the First Double Value:";
        cin>>val1;
        cout<<" Enter the Second Double Value:";
        cin>>val2;
        cout<<" Enter the Third Double Value:";
        cin>>val3;
        Arithmetic obj1(no1,no2,no3,str,str1,str2,val1,val2,val3);
        cout<<"\n Addition of Two Numbers is :"<<obj1.Add(no1,no2);
        cout<<"\n Addition of Three Numbers is :"<<obj1.Add(no1,no2,no3);
        cout<<"\n Addition of Two Strings :"<<obj1.Add(str,str1);
        cout<<"\n Addition of Three Strings :"<<obj1.Add(str,str1,str2);
        cout<<"\n Addition of Two Double values:"<<obj1.Add(val1,val2);
        cout<<"\n Addition of Three Double values:"<<obj1.Add(val1,val2,val3);
getch();
}
```

OUTPUT



RESULT

Thus, the program implements the function overloading for Add() with six different signatures.

IMPLEMENTING OPERATOR OVERLOADING

Expt.No-6 **Date: 17-08-2021**

AIM:

To implement operator overloading for Complex numbers using C++ Program

PROCEDURE:

Step-1	**Start** the Program
Step-2	**Declare** the class Complex
Step-3	**Declare** the class members real,imag as float
Step-4	**Create** a **Default Constructor** and initialize the members as 0.0
Step-5	**Create** an object C1 and C2 for Complex Class and initialize it using **Parameterized Constructor**.
Step-6	**Create** an object C3, C4, C5, C6 and initialize it using **Default Constructor**
Step-7	**Call operator + ()** as a member function and perform Addition of two complex numbers.
Step-8	**Call operator - ()** as a friend function and perform subtraction of two complex numbers
Step-9	**Call operator * ()** as a member function and perform multiplication of two complex numbers
Step-10	**Call operator / ()** as a friend function and perform Division of two complex numbers
Step-11	**Display** the result values.
Step-12	**END of the Program**

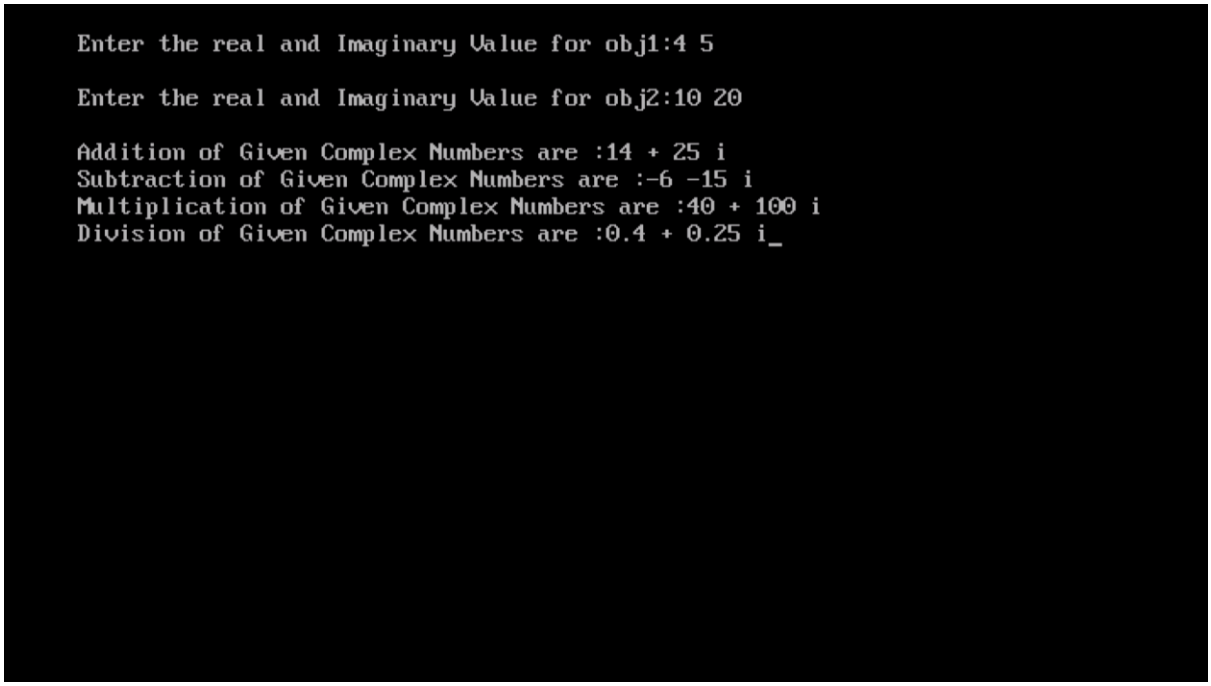
PROGRAM

```
 #include<iostream.h>
#include<conio.h>
class Complex
{
        float real, imag;
public:
        Complex()
        {
                real=imag=0.0;
        }
        Complex(float r, float i)
        {
                real= r;
                imag=i;
        }
        void Show_data()
        {
                if (imag<0)
                        cout<<real<<" "<<imag<<" i";
                else
                        cout<<real<<" + "<<imag<<" i";
        }
        Complex operator +(Complex &obj)
        {
                Complex temp;
                temp.real= real+obj.real;
                temp.imag= imag+obj.imag;
                return(temp);
        }
```

```
        friend Complex operator -(Complex obj1, Complex obj2)
        {
                Complex temp;
                temp.real= obj1.real-obj2.real;
                temp.imag= obj1.imag-obj2.imag;
                return(temp);
        }
        Complex operator * (Complex &);
        friend Complex operator / (Complex,Complex);
};
Complex Complex:: operator *(Complex &obj)
{
        Complex temp;
        temp.real=real*obj.real;
        temp.imag=imag*obj.imag;
        return(temp);
}
Complex operator /(Complex obj1, Complex obj2)
{
        Complex temp;
        temp.real=obj1.real/obj2.real;
        temp.imag= obj1.imag/obj2.imag;
        return(temp);
}
void main()
{
        float rvalue1,rvalue2,ivalue1,ivalue2;
clrscr();
        cout<<"\n Enter the real and Imaginary Value for obj1:";
        cin>>rvalue1>>ivalue1;
```

```
        cout<<"\n Enter the real and Imaginary Value for obj2:";
        cin>>rvalue2>>ivalue2;
        Complex C1(rvalue1,ivalue1),C2(rvalue2,ivalue2),C3,C4,C5,C6;
        C3= C1+C2;
        C4= C1-C2;
        C5= C1*C2;
        C6= C1/C2;
        cout<<"\n Addition of Given Complex Numbers are :";
        C3.Show_data();
        cout<<"\n Subtraction of Given Complex Numbers are :";
        C4.Show_data();
        cout<<"\n Multiplication of Given Complex Numbers are :";
        C5.Show_data();
        cout<<"\n Division of Given Complex Numbers are :";
        C6.Show_data();
getch();
}
```

OUTPUT



RESULT

Thus, the program implements the operator overloading for Complex class () using member functions and friend functions and perform arithmetic operations.

IMPLEMENTING INHERITANCE

Expt.No-7 **Date: 15-09-2021**

AIM:

To implement the concept of inheritance using C++ Program

PROGRAM

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
class College_details
{
        char college_name[75];
        char college_address[75];
        char university[50];
        char principal[50];
        char contact_number[15];
public:
        College_details()
        {
                cout<<" Enter the College_name: ";
                cin>>college_name;
                cout<<" Enter the College Address:";
                cin>>college_address;
                cout<<" Enter the University :       ";
                cin>>university;
                cout<<" Enter the Principal Name :";
                cin>>principal;
                cout<<" Enter the College Contact Number :";
                cin>>contact_number;
        }
```

```
        void Show_data()
        {
                cout<<"\n College Name: "<<college_name;
                cout<<"\n College Address : "<<college_address;
                cout<<"\n University :"<< university;
                cout<<"\n Principal :"<<principal;
                cout<<"\n Contact Number:"<<contact_number;
        }
        char* get_college()
        {
                return(college_name);
        }
        char* get_univ()
        {
                return(university);
        }
};
class College_dept_details: public College_details
{
        int dept_id;
        char depart_name[30];
        char course[30];
        char hod[50];
        char contact_no[15];
public:
        College_dept_details()
        {
                cout<<" Enter the Department ID:";
                cin>>dept_id;
                cout<<" Enter the Department Name:";
```

```
                cin>>depart_name;
                cout<<" Enter the course name :";
                cin>>course;
                cout<<" Enter the HOD Name :";
                cin>>hod;
                cout<<" Enter the Department contact No:";
                cin>>contact_no;
        }
        void Show_data()
        {       College_details::Show_data();
                cout<<"\n Department ID:"<<dept_id;
                cout<<"\n Department Name:"<<depart_name;
                cout<<"\n Course :"<<course;
                cout<<"\n HOD :"<<hod;
                cout<<"\n Contact No:"<<contact_no;
        }
        char * get_course()
        {
                return(course);
        }
        char* get_dept()
        {
                return(depart_name);
        }
};
class College_staff_details : public College_dept_details    // Multi level
Inheritance
{
        int staff_id;
        char staff_name[50];
        char qualification[50];
```

```
        char dept[50];
public:
        College_staff_details() // Constructor
        {
                cout<<" Enter the Staff Id:";
                cin>>staff_id;
                cout<<" Enter the Staff Name :";
                cin>>staff_name;
                cout<<" Enter the Staff Qualification:";
                cin>>qualification;
                strcpy(dept,get_dept());
        }
        void Show_data()
        {
                cout<<"\n Staff ID: "<<staff_id;
                cout<<"\n Staff Name:"<<staff_name;
                cout<<"\n Staff Qualification: "<<qualification;
                cout<<"\n Department ID :"<<dept;
        }
        char* get_staff()
        {
                return(staff_name);
        }
};
class Subject_handling: public College_staff_details
{
        char batch[15];
        char course[30];
        char subject[50];
        char staff[50];
```

```
public:
        Subject_handling()
        {
                cout<<" Enter the Batch:";
                cin>>batch;
                strcpy(course,get_course());
                cout<<" Enter the Subject:";
                cin>>subject;
                strcpy(staff,get_staff());
        }
        void Show_data()
        {
                College_staff_details::Show_data();
                cout<<"\n Batch:"<<batch;
                cout<<"\n Course:"<<course;
                cout<<"\n Subject:"<<subject;
                cout<<"\n Staff Name:"<<staff;
        }
        char* get_batch()
        {
                return(batch);
        }
};
class Student_profile_details // Base Class
{
        int stud_no;
        char stud_name[50];
        char f_name[50];
        char m_name[50];
        char dob[20];
```

```
        char aadharno[13];
        char address[75];
public:
        Student_profile_details()
        {
                cout<<"\n Enter the Student No:";
                cin>>stud_no;
                cout<<" Enter the Student Name:";
                cin>>stud_name;
                cout<<" Enter the Father Name:";
                cin>>f_name;
                cout<<" Enter the Mother Name:";
                cin>>m_name;
                cout<<" Enter the Date of Birth:";
                cin>>dob;
                cout<<" Enter the Student Aadhar no:";
                cin>>aadharno;
                cout<<" Enter the Student Address:";
                cin>>address;
        }
        void Show_data()
        {

                cout<<"\n Student No:"<<stud_no;
                cout<<"\n Student Name:"<<stud_name;
                cout<<"\n Father Name:"<<f_name;
                cout<<"\n Mother Name:"<<m_name;
                cout<<"\n Date of Birth:"<<dob;
                cout<<"\n Student Aadhar no:"<<aadharno;
                cout<<"\n Student Address:"<<address;
```

```
        }
        int get_studno()
        {
                return(stud_no);
        }
        char* get_studname()
        {
                return(stud_name);
        }
};
```

class Student_course_details: public Student_profile_details, public Subject_handling

```
{
        int stud_no;
        char course[20];
        char dept[20];
        char batch[20];
public:
        Student_course_details()
        {
                stud_no= get_studno();
                strcpy(course,get_course());
                strcpy(dept,get_dept());
                strcpy(batch,get_batch());
        }
        void Show_data()
        {
                cout<<"\n Student No:"<<stud_no;
                cout<<"\n Course :"<<course;
                cout<<"\n Department:"<<dept;
                cout<<"\n Batch:"<<batch;
```

```
        }
};
class Student_carrier_details: public Student_profile_details
{
        int stud_no;
        int sslc_rollno;
        int sslc_marks;
        char sslc_school[30];
        int hsc_rollno;
        int hsc_marks;
        char hsc_school[30];
public:
        Student_carrier_details()
        {
                stud_no= get_studno();
                cout<<"\n Enter the SSLC Roll no";
                cin>>sslc_rollno;
                cout<<" Enter the SSLC Marks:";
                cin>>sslc_marks;
                cout<<" Enter the SSLC School:";
                cin>>hsc_school;
                cout<<" Enter the HSC Roll no";
                cin>>hsc_rollno;
                cout<<" Enter the HSC Marks:";
                cin>>hsc_marks;
                cout<<" Enter the HSC School:";
                cin>>hsc_school;
        }
        void Show_data()
        {
```

```
                cout<<" Student No: "<<stud_no;
                cout<<"\n Enter the SSLC Roll no"<<sslc_rollno;
                cout<<"\n Enter the SSLC Marks:"<<sslc_marks;
                cout<<"\n Enter the SSLC School:"<<hsc_school;
                cout<<"\n Enter the HSC Roll no"<<hsc_rollno;
                cout<<"\n Enter the HSC Marks:"<<hsc_marks;
                cout<<"\n Enter the HSC School:"<<hsc_school;
        }
};
class Student_contact_details: public Student_profile_details
{
        int stud_no;
        char stud_mobile_no[15];
        char f_mobile_no[15];
        char m_mobile_no[15];
public:
        Student_contact_details()
        {
                stud_no= get_studno();
                cout<<"\n Enter the Student Mobile No:";
                cin>>stud_mobile_no;
                cout<<" Enter the Father Mobile No:";
                cin>>f_mobile_no;
                cout<<" Enter the Mother Mobile No :";
                cin>>m_mobile_no;
        }
        void Show_data()
        {
                cout<<"\n Student Roll No:"<<stud_no;
                cout<<"\n Student Mobile No:"<<stud_mobile_no;
```

```
                cout<<"\n Father Mobile No:"<<f_mobile_no;
                cout<<"\n Mother Mobile No :"<<m_mobile_no;
        }
};
class Student_semester_marks: public Student_course_details
{
        int stud_no;
        int sem;
        int marks[5];
        int tot;
        char result[10];
public:
        void Show_data();
        Student_semester_marks();
};
Student_semester_marks::Student_semester_marks()
{
        tot=0;
        strcpy(result,"Pass");
        stud_no= get_studno();
        cout<<" Enter the Semester :";
        cin>>sem;
        for(int i=0;i<5;i++)
        {
                cout<<" Enter the Marks:";
                cin>>marks[i];
                tot+= marks[i];
                if(marks[i]<35)
                {
                        strcpy(result,"Fail");
```

```
                }
        }
}
void Student_semester_marks:: Show_data()
{
                cout<<"\n Stud No    : "<<stud_no;
                cout<<"\n Stud Name  : "<<get_studname();
                cout<<"\n Course     : "<<get_course();
                cout<<"\n Department : "<<get_dept();
                cout<<"\n College    : "<<get_college();
                cout<<"\n University : "<<get_univ();
                cout<<"\n Semester   : "<<sem;
                for(int i=0;i<5;i++)
                {
                        cout<<"\n Marks      : "<<marks[i];
                }
                cout<<"\n Total      : "<<tot<<"\n Result     : "<<result;
}
void main()
{
clrscr();
        Student_semester_marks sm;
getch();
clrscr();
        sm.Show_data();
getch();
}
```

OUTPUT

```
Enter the Student No:21312
Enter the Student Name:Praveen
Enter the Father Name:Sundar
Enter the Mother Name:Sundari
Enter the Date of Birth:01-01-2003
Enter the Student Aadhar no:437024605345
Enter the Student Address:Vellore
Enter the College_name: APCAS
Enter the College Address:Kalavai
Enter the University : Thiruvalluvar
Enter the Principal Name :Sadiq
Enter the College Contact Number :9884398843
Enter the Department ID:25
Enter the Department Name:CS
Enter the Course name :BCA
Enter the HOD Name :Anandaraj
Enter the Department contact No:9884098840
Enter the Staff Id:05
Enter the Staff Name :Sankar
Enter the Staff Qualification:MCA
Enter the Batch:2020-23
Enter the Subject:C++
Enter the Semester :3_
```

```
Enter the Marks:60
Enter the Marks:70
Enter the Marks:80
Enter the Marks:90
Enter the Marks:95
_
```

```
Stud No    : 21312
Stud Name  : Praveen
Course     : BCA
Department : CS
College    : APCAS
University : Thiruvalluvar
Semester   : 3
Marks      : 60
Marks      : 70
Marks      : 80
Marks      : 90
Marks      : 95
Total      : 395
Result     : Pass
```

RESULT

Thus, the program implements the concept of Inheritance to maintain Student details.

IMPLEMENTING VIRTUAL FUNCTIONS

Expt.No-8 **Date: 28-09-2021**

AIM:

To implement virtual functions to display odd or even numbers using C++ Program

PROCEDURE:

Step-1	**Start** the Program
Step-2	**Declare** the class Numbers
Step-3	**Declare** the member function print_numbers as **virtual**
Step-4	**Declare** the class Odd_numbers which inherits abstract class Numbers
Step-5	**Define** the virtual function print_numbers in derived class Odd_numbers
Step-6	**Declare** the class Even_numbers which inherits abstract class Numbers
Step-7	**Define** the virtual function print_numbers in derived class Even_numbers
Step-8	**Declare** the pointer object to Abstract class Numbers
Step-9	**Assign** the Odd_numbers object to Numbers Class
Step-10	**Assign** the even_numbers object to Numbers Class
Step-11	**Display** the result values.
Step-12	**END of the Program**

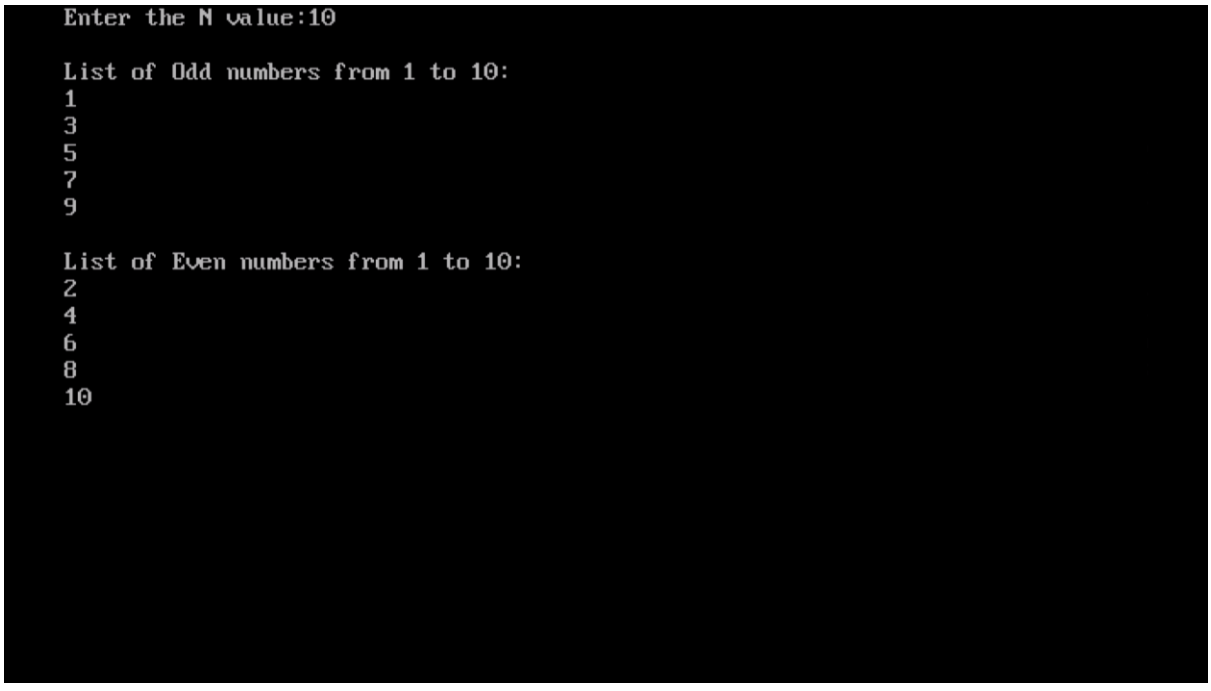
PROGRAM

```
#include<iostream.h>
#include<conio.h>
class Numbers
{
public:
        virtual void print_numbers(int)=0;
};
```

```
class Odd_numbers:public Numbers
{
public:
	void print_numbers(int);
};
void Odd_numbers::print_numbers(int n)
{
	cout<<"\nList of Odd numbers from 1 to "<<n<<":\n";
	for(int i=1;i<=n;i+=2)
	{
		cout<<i<<endl;
	}
}
class Even_numbers:public Numbers
{
public:
	void print_numbers(int);
};
void Even_numbers::print_numbers(int n)
{
	cout<<"\nList of Even numbers from 1 to "<<n<<":\n";
	for(int i=2;i<=n;i+=2)
	{
		cout<<i<<"\n";
	}
}
void main()
{
clrscr();
	int n;
```

```
        cout<<"Enter the N value:";
        cin>>n;
        Odd_numbers on;
        Even_numbers en;
        Numbers *ptr;
        ptr= &on;
        ptr->print_numbers(n);
        ptr=&en;
        ptr->print_numbers(n);
getch();
}
```

OUTPUT



RESULT

Thus, the program implements the pure virtual functions to the abstract class.

IMPLEMENTING FILE HANDLING CONCEPTS

Expt.No-9 **Date: 13-09-2021**

AIM:

To maintain Stock details in files using C++ Program

PROCEDURE:

Step-1	**Start** the Program
Step-2	**Declare** the class Inventory
Step-3	**Declare** the class member prod_name as char array
Step-4	**Declare** the class member prod_code as integer
Step-5	**Declare** the class member cost as float
Step-6	**Create** Object for Inventory class
Step-7	**Call** read_data() to get inputted values from console and to store the same in file.
Step-8	**Call** write_data() to get values from file and display the values to console.
Step-9	**Display** the File contents using read().
Step-10	**END of the Program**

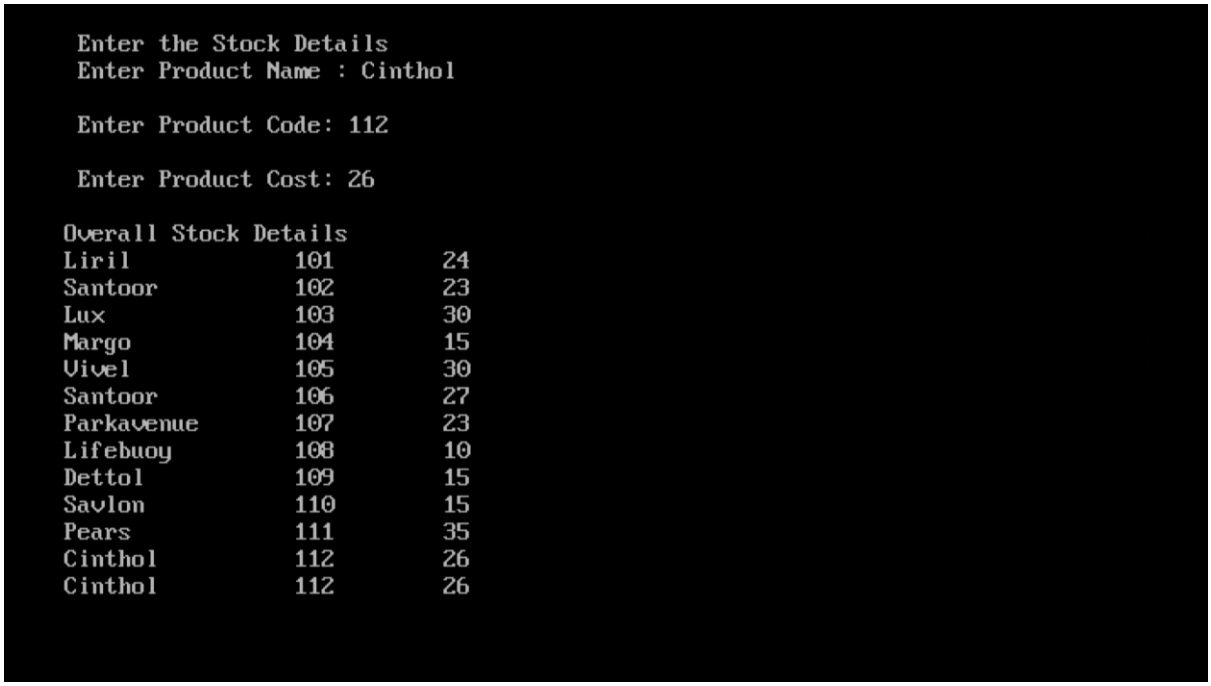
PROGRAM

```
 #include<iostream.h>
#include<conio.h>
#include<fstream.h>
#include<iomanip.h>
class Inventory
{
        char prod_name[20];
        int prod_code;
        float cost;
public:
```

```
        void read_data()
        {
                cout<<"\n Enter Product Name : ";
                cin>>prod_name;
                cout<<"\n Enter Product Code: ";
                cin>>prod_code;
                cout<<"\n Enter Product Cost: ";
                cin>>cost;
        }
        void write_data()
        {
                cout<<setiosflags(ios::left)
                   <<setw(10)<<prod_name
                   <<setiosflags(ios::right)
                   <<setw(10)<<prod_code
                   <<setprecision(2)
                   <<setw(10)<<cost<<endl;
        }
};
void main()
{
        Inventory item;
        int i;
clrscr();
        fstream file;
        file.open("stock.txt",ios::in|ios::app);
        cout<<"\n Enter the Stock Details";
        item.read_data();
        file.write((char*) &item,sizeof(item));
        file.seekg(0);
```

```
        cout<<"\nOverall Stock Details\n";
        while(file)
        {
                file.read((char *) &item,sizeof(item));
                item.write_data();
        }
        file.close();
getch();
}
```

OUTPUT



RESULT

Thus, the program implements File concepts to maintain Stock details.

STRING HANDLING FUNCTIONS

Expt.No-10 **Date: 20-09-2021**

AIM:

To perform string handling functions which are available in <string.h>.

PROCEDURE:

Step-1	**Start** the Program
Step-2	**Declare** the string variables str,newstr,second_Str as character array
Step-3	**Get** the Length of string using strlen()
Step-4	**Convert** the given string into Lower Case using strlwr()
Step-5	**Convert** the given string into Upper Case using strupr()
Step-6	**Copy** the Given String into newstr and **Concat** the both strings
Step-7	**Reverse** the Given strings
Step-8	**Compare** the Both Strings for Palindrome using strcmp()
Step-9	**Display** the Results
Step-10	**END of the Program**

PROGRAM

```
 #include<iostream.h>
#include<conio.h>
#include<string.h>
void main()
{
        char str[100],newstr[100],second_str[100];
clrscr();
        cout<<"\n Enter the String: ";
        cin>>str;
        cout<<" Enter the Second String : ";
```

```
        cin>>second_str;
        cout<<" Length of the String : "<<strlen(str)<<endl;
        cout<<" Lowercase of the String : "<<strlwr(str)<<endl;
        cout<<" Uppercase of the String : "<<strupr(str)<<endl;
        cout<<" Length of the Second String : "<<strlen(second_str)<<endl;
        cout<<" Lowercase of the Second String : "<<strlwr(second_str)<<endl;
        cout<<" Uppercase of the Second String : "<<strupr(second_str)<<endl;
        cout<<" Reverse of the Given String : "<<strrev(str)<<endl;
        cout<<" Reverse of the Given Second String : "<<strrev(second_str);
        if(strcmp(str,second_str)==0)
        {
                cout<<"\n Given String is Palindrome ";
        }
        else
        {
                cout<<"\n Given String is not Palindrome ";
        }
        strcpy(newstr,str);
        strcat(newstr," ");
        strcat(newstr,second_str);
        cout<<"\n New String after Concatination : "<<newstr<<endl;
getch();
}
```

OUTPUT

```
Enter the String: liril
Enter the Second String : liril
Length of the String : 5
Lowercase of the String : liril
Uppercase of the String : LIRIL
Length of the Second String : 5
Lowercase of the Second String : liril
Uppercase of the Second String : LIRIL
Reverse of the Given String : LIRIL
Reverse of the Given Second String : LIRIL
Given String is Palindrome
New String after Concatination : LIRIL LIRIL
```

```
Enter the String: Praveen
Enter the Second String : Sundar
Length of the String : 7
Lowercase of the String : praveen
Uppercase of the String : PRAVEEN
Length of the Second String : 6
Lowercase of the Second String : sundar
Uppercase of the Second String : SUNDAR
Reverse of the Given String : NEEVARP
Reverse of the Given Second String : RADNUS
Given String is not Palindrome
New String after Concatination : NEEVARP RADNUS
```

RESULT

Thus, the program implements String handling functions.