PRACTICAL FILE
ON
OBJECT ORIENTED PROGRAMMING USING C++
[ES203]

SUBMITTED TO:	**SUBMITTED BY:**
Ms. Swati Singal	**Name: Raunaq Duggal**
ASSOCIATE PROFESSOR	**Enrolment No.: A2305221111**
	B.Tech. 3CSE6X

COMPUTER SCIENCE AND ENGINEERING
AMITY SCHOOL OF ENGINEERING &TECHNOLOGY
AMITY UNIVERSITY, UTTAR PRADESH
SESSION- 2022-2023

PRACTICAL 1

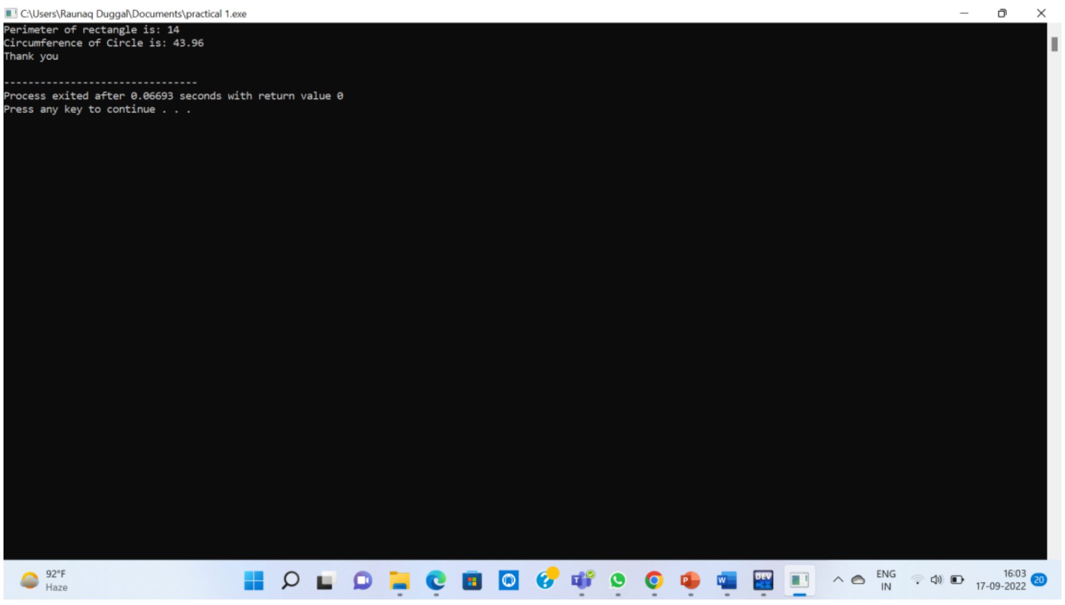
Aim-Define a class Shape whose attributes are radius, length and width calculate the perimeter of the rectangle and circle. Use constructors and destructors.

Software Used-Dev C++

Program-

```
#include<iostream>
using namespace std;
class Shape
{
   int length,width;
   float radius;

   public:

   Shape(int length,int width,float radius)
   {
      cout << "Perimeter of rectangle is: " << 2*(length+width) << endl;
      cout << "Circumference of Circle is: " << 2*3.14*radius << endl;
   };

   ~Shape()
   {
      cout << "Thank you" << endl;
   };

};

int main()
{
   Shape S1(2,5,7);
   return 0;

}
```

Output-



```
Perimeter of rectangle is: 14
Circumference of Circle is: 43.96
Thank you

--------------------------------
Process exited after 0.06693 seconds with return value 0
Press any key to continue . . .
```

Conclusion- The concept of constructors and destructors in C++ has been studied.

Remarks-

Signature-

PRACTICAL 2

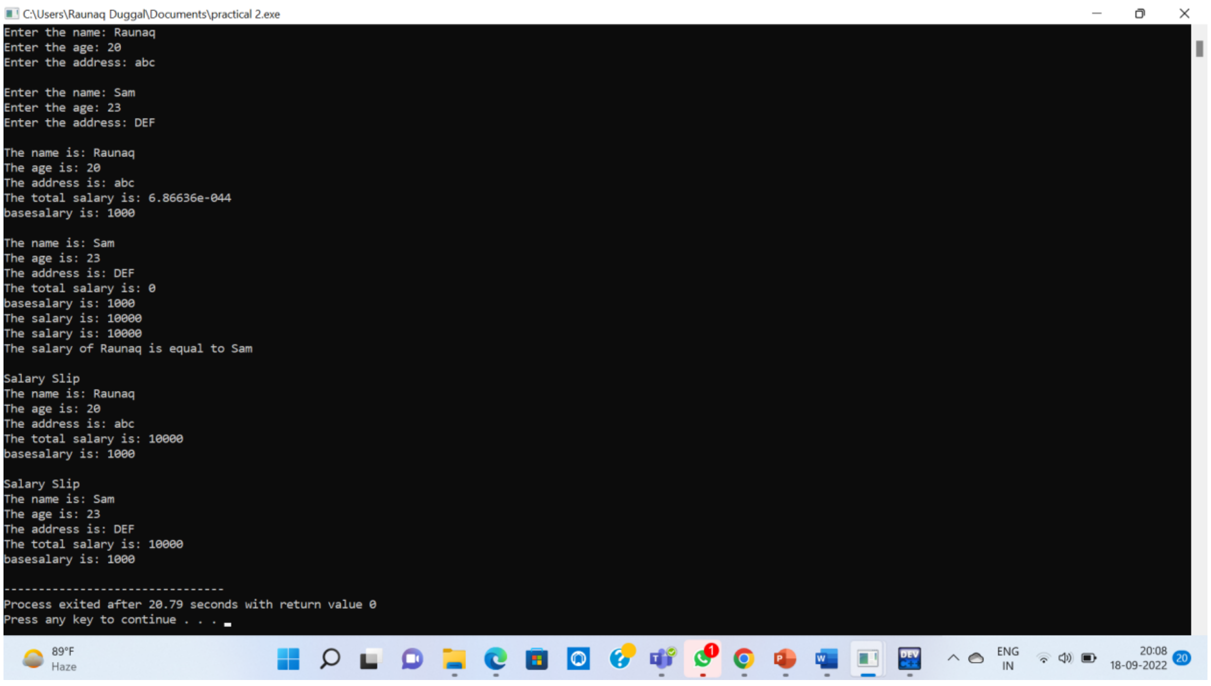
-
Aim-To create a class Person which includes: character array name of size 64, age in numeric, character array address of size 64, and total salary in real numbers (divide salary in different components, if required). Make an array of objects of class Person of size 10.
(a) Write inline function that obtains the youngest and eldest age of a person from a class person using arrays.
(b) Write a program to develop the salary slip and display result by using constructors.

Software Used-Dev C++

Program-
```
#include<iostream>
using namespace std;
class Person
{
    char name1[64], name2[64];
    int age1, age2;
    char address1[64], address2[64];
    float totalSalary1, totalSalary2;
    int basesalary;

    public:

        void GetPersonData()
        {
            cout << "Enter the name: ";
            cin >> name1;
            cout << "Enter the age: ";
            cin >> age1;
            cout << "Enter the address: ";
            cin >> address1;
            cout << "\nEnter the name: ";
            cin >> name2;
            cout << "Enter the age: ";
            cin >> age2;
            cout << "Enter the address: ";
            cin >> address2;

        }
       void  DisplayPersonData()
        {
            cout << "\nThe name is: " << name1 << endl;
            cout << "The age is: " << age1 << endl;
            cout << "The address is: " << address1 << endl;
            cout << "The total salary is: " << totalSalary1 << endl;
            cout << "basesalary is: "<< basesalary <<endl;
            cout << "\nThe name is: " << name2 << endl;
            cout << "The age is: " << age2 << endl;
            cout << "The address is: " << address2 << endl;
            cout << "The total salary is: " << totalSalary2 << endl;
            cout << "basesalary is: "<< basesalary <<endl;
        }

        int ComputeSalary1()
        {
            if(age1 < 25)
            {
                totalSalary1 = 10000;
            }
            else if(age1 >= 25 && age1 <= 45)
            {
                totalSalary1 = 15000;
            }
            else
            {
                totalSalary1 = 20000;
            }
            cout << "The salary is: " << totalSalary1 << endl;
            return 0;
        }

        int ComputeSalary2()
        {
            if(age2 < 25)
            {
                totalSalary2 = 10000;
            }
            else if(age2 >= 25 && age2 <= 45)
            {
                totalSalary2 = 15000;
            }
            else
            {
                totalSalary2 = 20000;
            }
            cout << "The salary is: " << totalSalary2 << endl;
            return 0;
        }

        int Compare() //inline function
        {
            if (totalSalary1 > totalSalary2)
            {
                cout << "The salary of " << name1 << " is greater than " << name2 << endl;
            }
            else if(totalSalary1 < totalSalary2)
```

```
            {
               cout << "The salary of " << name2 << " is greater than " << name1 << endl;
            }
            else
            {
               cout << "The salary of " << name1 << " is equal to " << name2 << endl;

            }
         return 0;
         }
         Person() //Constructor

         {
            basesalary = 1000;
         }

      void DisplayData();

   };
      void Person :: DisplayData()
      {
            cout <<"\nSalary Slip"<< endl;
            cout << "The name is: " << name1 << endl;
            cout << "The age is: " << age1 << endl;
            cout << "The address is: " << address1 << endl;
            cout << "The total salary is: " << totalSalary1 << endl;
            cout << "basesalary is: "<< basesalary <<endl;

            cout <<"\nSalary Slip"<< endl;
            cout << "The name is: " << name2 << endl;
            cout << "The age is: " << age2 << endl;
            cout << "The address is: " << address2 << endl;
            cout << "The total salary is: " << totalSalary2 << endl;
            cout << "basesalary is: "<< basesalary <<endl;
      }

   int main()
   {
      Person P;
      P.GetPersonData();
      P.DisplayPersonData();
      P.ComputeSalary1();
      P.ComputeSalary2();
      P.Compare();
      P.DisplayData();

   }
```

Output-



C:\Users\Raunaq Duggal\Documents\practical 2.exe

```
Enter the name: Raunaq
Enter the age: 20
Enter the address: abc

Enter the name: Sam
Enter the age: 23
Enter the address: DEF

The name is: Raunaq
The age is: 20
The address is: abc
The total salary is: 6.86636e-044
basesalary is: 1000

The name is: Sam
The age is: 23
The address is: DEF
The total salary is: 0
basesalary is: 1000
The salary is: 10000
The salary is: 10000
The salary of Raunaq is equal to Sam

Salary Slip
The name is: Raunaq
The age is: 20
The address is: abc
The total salary is: 10000
basesalary is: 1000

Salary Slip
The name is: Sam
The age is: 23
The address is: DEF
The total salary is: 10000
basesalary is: 1000

--------------------------------
Process exited after 20.79 seconds with return value 0
Press any key to continue . . .
```

Conclusion-The concept of array of objects, inline function and constructors in C++ has been studied.

Remarks-

Signature-

PRACTICAL 3

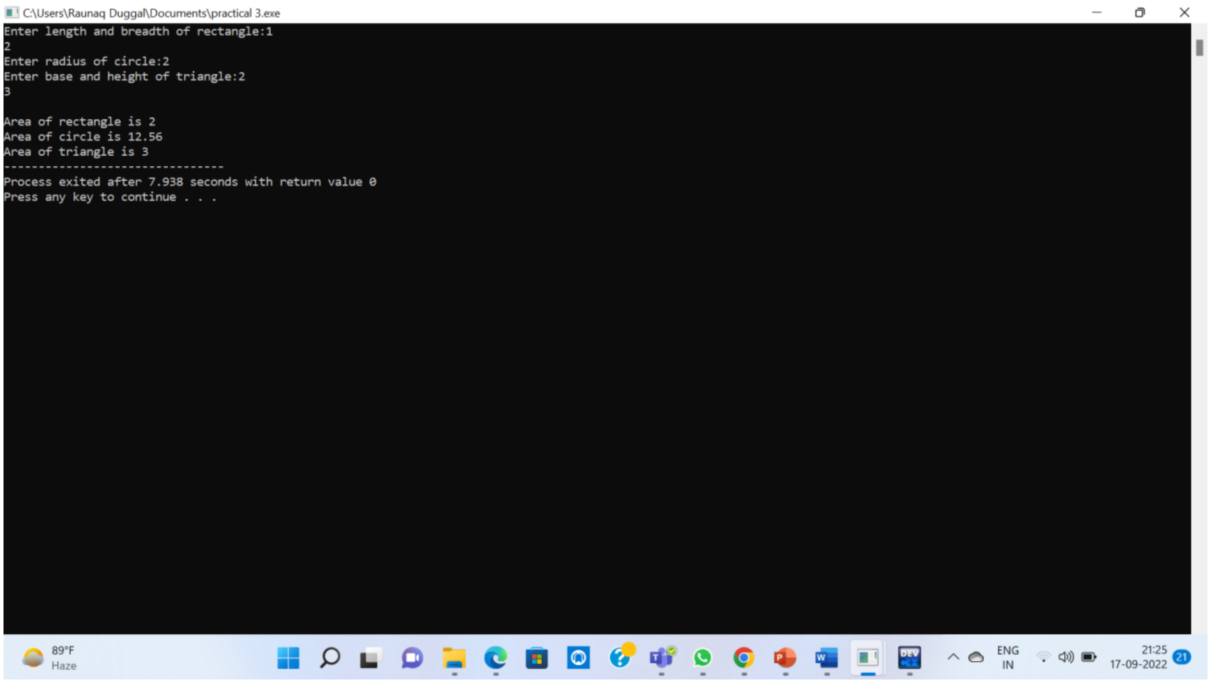
Aim-Write a program to find the area (function name AREA) of circle, rectangle and triangle by Function overloading concept.

Software Used-Dev C++

Program-

```
#include<iostream>
using namespace std;
int AREA(int,int);
float AREA(float);
float AREA(float,float);
int main()
{
	int l,b;
	float r,bs,ht;

	cout<<"Enter length and breadth of rectangle:";
	cin>>l>>b;
	cout<<"Enter radius of circle:";
	cin>>r;
	cout<<"Enter base and height of triangle:";
	cin>>bs>>ht;
	cout<<"\nArea of rectangle is "<<AREA(l,b);
	cout<<"\nArea of circle is "<<AREA(r);
	cout<<"\nArea of triangle is "<<AREA(bs,ht);
}

int AREA(int l,int b)
{
   return(l*b);
}
float AREA(float r)
{
   return(3.14*r*r);
}
float AREA(float bs,float ht)
{
  return((bs*ht)/2);
}
```

Output-



C:\Users\Raunaq Duggal\Documents\practical 3.exe

```
Enter length and breadth of rectangle:1
2
Enter radius of circle:2
Enter base and height of triangle:2
3

Area of rectangle is 2
Area of circle is 12.56
Area of triangle is 3
--------------------------------
Process exited after 7.938 seconds with return value 0
Press any key to continue . . .
```

Conclusion- The concept of Function overloading in C++ has been studied.

Remarks-

Signature-

PRACTICAL 4

Aim- a) Write a program to swap two numbers (create two classes) by using Friend function.
b)Write a program to create two classes DistM and DistF which store the values of distance. DistM stores distance in meters and centimetres and DistF stores distances in feet and inches. Read values for the class object and add one object of DistM with another object of DistF. Use a friend function for the addition operation and display answer in meter and centimetres.

Software Used- Dev C++

Program-

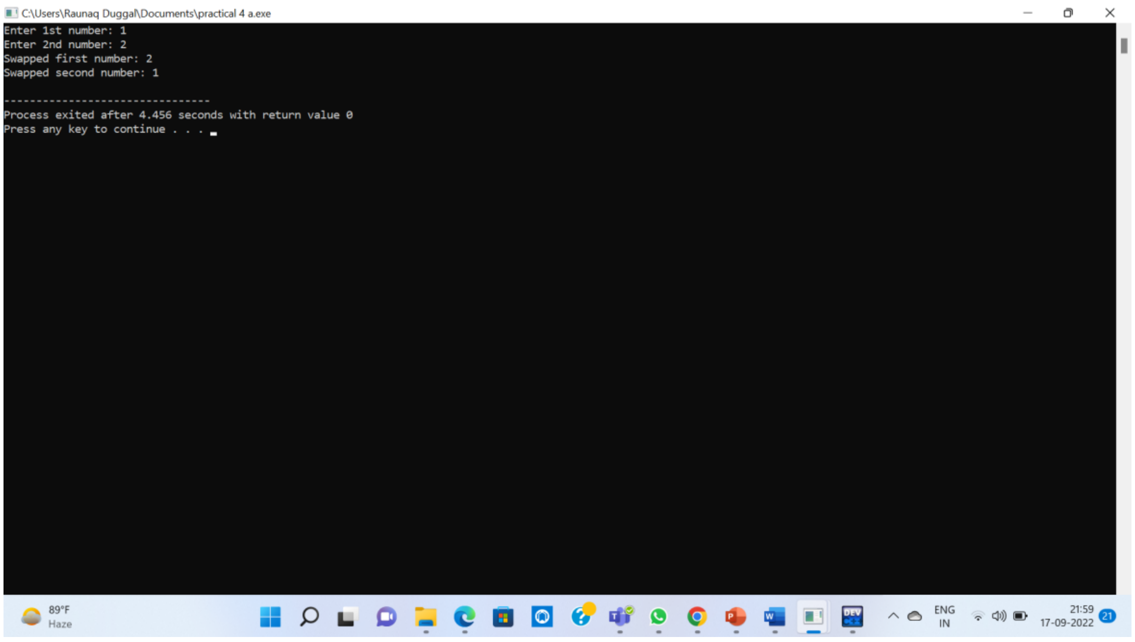
a)
```
#include<iostream>
using namespace std;
class Swap1
{
   int a,b
   public:
   friend void swap(int a,int b);
};

void swap(int a,int b)
{
   int temp = 0;
   temp=a;
   a=b;
   b=temp;
   cout << "Swapped first number: " << a<< endl;
   cout << "Swapped second number: " << b<< endl;
}


int main()
{
   int x,y;
   cout << "Enter 1st number: " ;
   cin >> x;
   cout << "Enter 2nd number: ";
   cin >> y;

   Swap1 S1;
   swap(x,y);
   return 0;
}
```

b) #include<iostream>
```
using namespace std;
class DistF;
class DistM
{
   float m;
   float cm;
   public:
   void getdata();
   void display();
   friend DistM add(DistM x, DistF y);
};

class DistF
{
   int feet;
   int inches;
   public:
   void getdata();
   void display();
   friend DistM add(DistM x, DistF y);
};

void DistM :: getdata()
{
   cout<< "Enter in meters: ";
   cin >> m;
   cout<< "Enter in centimeters: ";
   cin >> cm;
}

void DistM :: display()
{
   cout<< "Distance in meters: " << m << endl;
   cout<< "Distance in centimeters: " << cm << endl;
}

void DistF :: getdata()
{
   cout<< "Enter in feet: ";
   cin >> feet;
   cout<< "Enter in inches: ";
   cin>> inches;
}

void DistF :: display()
{
   cout << "Distance in feet: " << feet << endl;
```

```
    cout << "Distance in inches: " << inches << endl;
}

DistM add (DistM x, DistF y)
{
    DistM temp;
    DistF temp1;

    temp.m = x.m + x.cm/100 + y.inches + 0.0254 + y.feet*0.3048;
    temp.cm = temp.m*100;

    temp1.feet = temp.m*3.2808;
    temp1.inches = temp.m*39.37;

    return temp;
}

int main()
{
    DistM a;
    a.getdata();
    DistF b;
    b.getdata();
    DistM c;

    c = add(a,b);
    c.display();

    return 0;
}
```

Output-

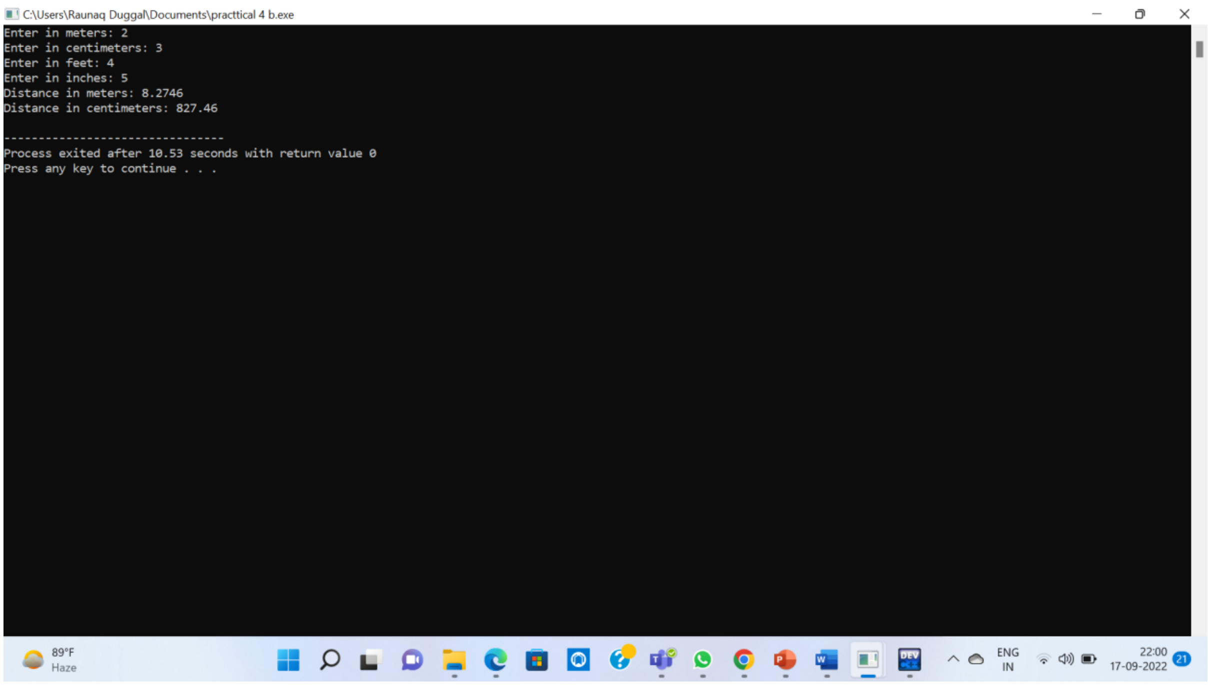
a)



C:\Users\Raunaq Duggal\Documents\practical 4 a.exe

```
Enter 1st number: 1
Enter 2nd number: 2
Swapped first number: 2
Swapped second number: 1

--------------------------------
Process exited after 4.456 seconds with return value 0
Press any key to continue . . .
```

b)



C:\Users\Raunaq Duggal\Documents\practtical 4 b.exe

```
Enter in meters: 2
Enter in centimeters: 3
Enter in feet: 4
Enter in inches: 5
Distance in meters: 8.2746
Distance in centimeters: 827.46

--------------------------------
Process exited after 10.53 seconds with return value 0
Press any key to continue . . .
```

Conclusion-The concept of friend function in C++ has been studied .

Remarks-

Signature-

PRACTICAL 5

Aim-Write a program in which length is measured in feet and inches. User enters two values of lengths then a menu will be displayed for performing the following operations on it. Use operator overloading for all the functions:

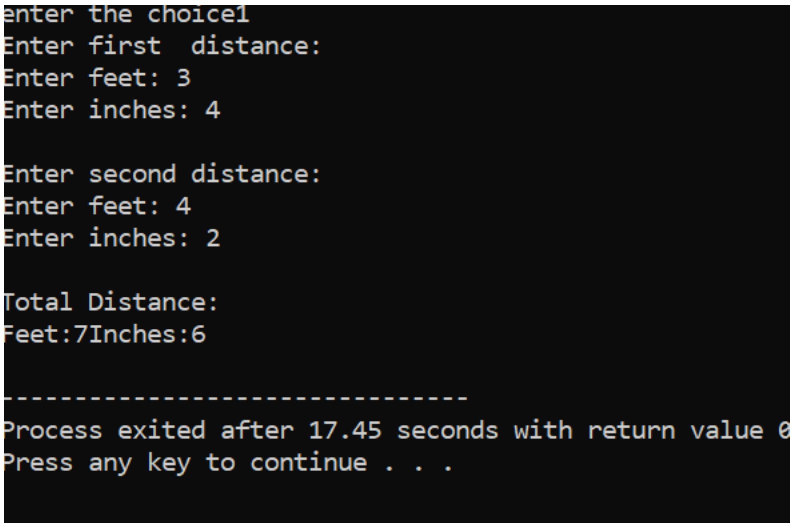
1. Add two lengths: + operator
2. Compare the lengths using < operator
3. Compare the lengths using == operator
4. Use *= operator to multiply the length with given integer value

Software Used-Dev C++

Program-

```
#include <iostream>
using namespace std;
class Distance
{
public:
   int feet, inches;
public:
   void setDistance(void)
   {
      cout << "Enter feet: ";
      cin >> feet;
      cout << "Enter inches: ";
      cin >> inches;
   }
   void displayDistance(void)
   {
      cout << "Feet:" << feet << "Inches:" << inches << endl;
   }
   Distance operator+(Distance& dist1)
   {
      Distance tempD;
      tempD.inches = inches + dist1.inches;
      tempD.feet = feet + dist1.feet + (tempD.inches / 12);
      tempD.inches = tempD.inches % 12;
      return tempD;
   }
   Distance operator<(Distance& dist1)
   {
      Distance tempD;
      tempD.inches = inches + dist1.inches;
      tempD.feet = feet + dist1.feet + (tempD.inches / 12);
      tempD.inches = tempD.inches % 12;
      return tempD;
   }
   Distance operator==(Distance& dist1)
   {
      Distance tempD;
      tempD.inches = inches + dist1.inches;
      tempD.feet = feet + dist1.feet + (tempD.inches / 12);
      tempD.inches = tempD.inches % 12;
      return tempD;
   }
   Distance operator*=(Distance& dist1)
   {
      Distance tempD;
      tempD.inches = inches + dist1.inches;
      tempD.feet = feet + dist1.feet + (tempD.inches / 12);
      tempD.inches = tempD.inches % 12;
      int x;
      cout<<"enter the integer:";
      cin>>x;
      feet*=x;
      inches*=x;
      cout<<"\n\nthe multiplied distance is:"<<feet<<"  "<<inches;
      cout<<"\n";
      return tempD;
   }
};
int main()
{
   int choice;
   cout<<"enter the choice";
   cin>>choice;
   Distance D1, D2, D3,D4,D5;
   cout << "Enter first  distance:" << endl;
   D1.setDistance();
   cout << endl;
   cout << "Enter second distance:" << endl;
   D2.setDistance();
   cout << endl;
   switch(choice)
   {
      case 1:
      D3 = D1 + D2;
      cout << "Total Distance:" << endl;
      D3.displayDistance();
      break;
      case 2:

if(D1.feet<D2.feet&&D1.inches<D2.inches||D1.feet<D2.feet&&D2.inches>D1.inches||D2.fe
et<D1.feet&&D2.inches<D1.inches)
      {
          D2.displayDistance();
          cout<<"they are greater";
      }
      else
      {
          D1.displayDistance();
          cout<<"they are greater";
      }
      break;
      case 3:
      if(D1.feet==D2.feet && D1.inches==D2.inches )
      {
          cout<<"they are equal";
      }
      else
      {
          cout<<"they are not equal";
      }
      break;
      case 4:
      D4=D1*=D2;
      D5=D2*=D1;
      break;
      default:
      cout<<"invalid";
   }
   return 0;
}
```

Output-



```
enter the choice1
Enter first  distance:
Enter feet: 3
Enter inches: 4

Enter second distance:
Enter feet: 4
Enter inches: 2

Total Distance:
Feet:7Inches:6

--------------------------------
Process exited after 17.45 seconds with return value 0
Press any key to continue . . .
```

Conclusion-The concept of operator overloading in C++ has been studied.

Remarks-

Signature-

PRACTICAL 6

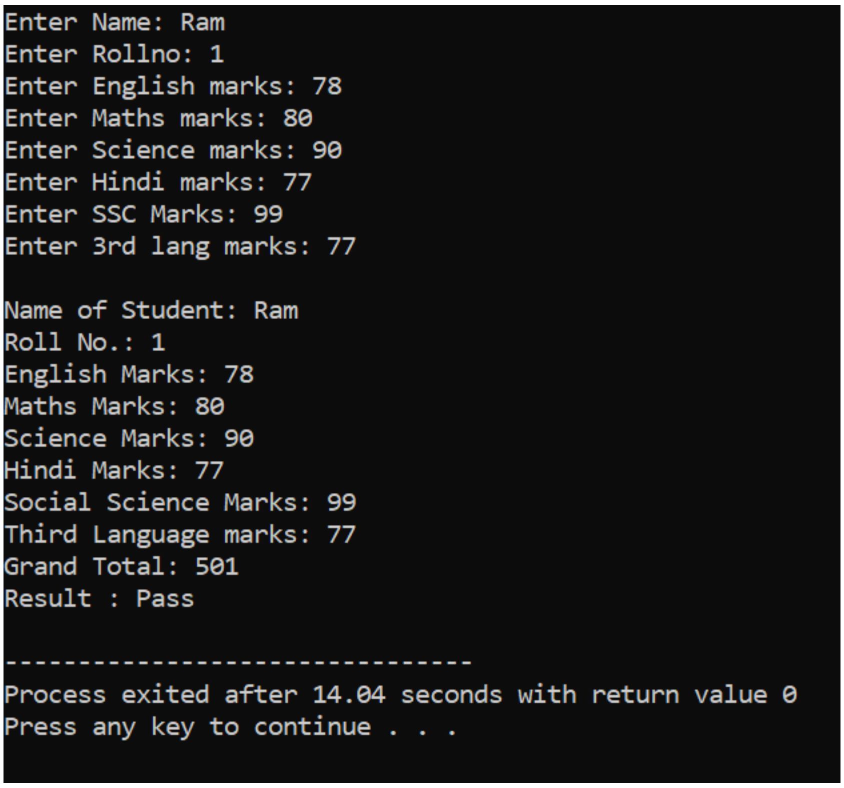
Aim-Design three classes: Student, Exam and Result. The student class has data members such as roll no, name etc. Create a class Exam by inheriting the Student class. The Exam class adds data members representing the marks scored in six subjects. Derive the Result from class Exam and it has its own members such as total marks. Write an interactive program to model this relationship. What type of **inheritance** this model belongs to?

Software Used- Dev C++

Program-

```
#include<iostream>
using namespace std;
class Student
{
 protected:
 int Rollno;
 char name[20];
 public:
 void getdata()
 {
 cout << "Enter Name: ";
 cin >> name;
 cout << "Enter Rollno: ";
 cin >> Rollno;
 }
};
class Exam : public Student
{
 protected:
 int engmarks;
 int mathsmarks;
 int scmarks;
 int hindimarks;
 int sscmarks;
 int thlangmarks;
 public:
 void getmarks()
 {
 cout << "Enter English marks: ";
 cin >> engmarks;
 cout << "Enter Maths marks: ";
 cin >> mathsmarks;
 cout << "Enter Science marks: ";
 cin >> scmarks;
 cout << "Enter Hindi marks: ";
 cin >> hindimarks;
 cout << "Enter SSC Marks: ";
 cin >> sscmarks;
 cout << "Enter 3rd lang marks: ";
 cin >> thlangmarks;
 }
};
class Result : public Exam
{
 protected:
 int total;
 public:
 void totalmarks()
 {
 total = engmarks + mathsmarks + scmarks + hindimarks + sscmarks + thlangmarks;
 }
 void display();
};
void Result :: display()
{
 cout << endl;
 cout << "Name of Student: " << name <<endl;
 cout << "Roll No.: " << Rollno << endl;
 cout << "English Marks: " << engmarks << endl;
 cout << "Maths Marks: " << mathsmarks << endl;
 cout << "Science Marks: " << scmarks << endl;
 cout << "Hindi Marks: " << hindimarks << endl;
 cout << "Social Science Marks: " << sscmarks << endl;
 cout << "Third Language marks: " << thlangmarks << endl;
 cout << "Grand Total: " << total << endl;
 if (total < 198)
 {
 cout << "Result : Fail" << endl;
 }
 else
 {
 cout << "Result : Pass" << endl;
 }
}
int main()
{
 Result R1;
 R1.getdata();
 R1.getmarks();
 R1.totalmarks();
 R1.display();
 return 0;
}
```

<u>Output-</u>



```
Enter Name: Ram
Enter Rollno: 1
Enter English marks: 78
Enter Maths marks: 80
Enter Science marks: 90
Enter Hindi marks: 77
Enter SSC Marks: 99
Enter 3rd lang marks: 77

Name of Student: Ram
Roll No.: 1
English Marks: 78
Maths Marks: 80
Science Marks: 90
Hindi Marks: 77
Social Science Marks: 99
Third Language marks: 77
Grand Total: 501
Result : Pass

--------------------------------
Process exited after 14.04 seconds with return value 0
Press any key to continue . . .
```

<u>Conclusion</u>-The concept of inheritance in C++ has been studied.

<u>Remarks-</u>

<u>Signature-</u>

PRACTICAL7

Aim-Consider an example of book shop which sells books and video tapes. These two classes are inherited from base class called media. The media class has command data members such as title and publication. The book class has data members for storing number of pages in a book and tape class has playing time in a tape. Each class will have member functions such as read () and show (). In the base class, these members have to be defined as **virtual functions**. Write a program to models the class hierarchy for book shop and processes objects of these classes using pointers to base class. Write the rules of virtual functions.

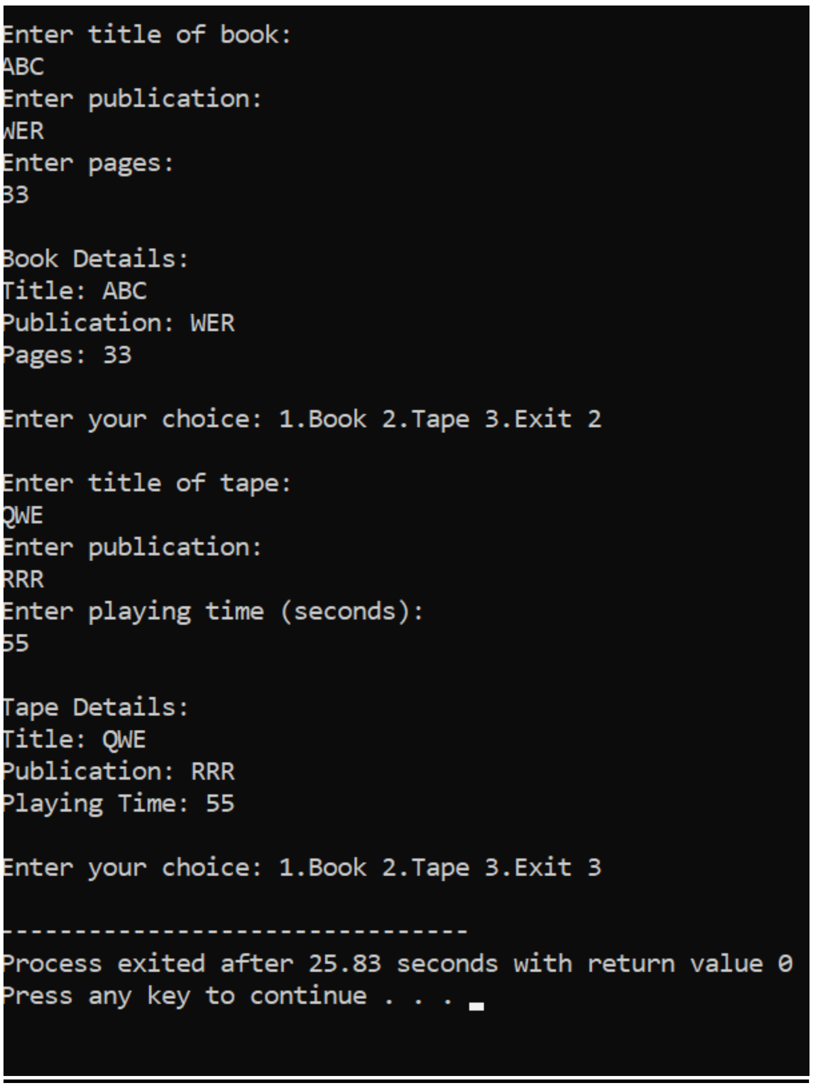
Software Used-Dev C++

Program-

```
#include <iostream>
using namespace std;

class media
{
protected:
char title[30];
char publication[30];
public:
virtual void read()=0;
virtual void show()=0;
};

class book:public media
{
int pages;
public:
void read();
void show();
};

class tape:public media
{
int time;
public:
void read();
void show();
};

void book::read()
{
cout<<"\nEnter title of book: "<<endl;
cin>>title;
cout<<"Enter publication: "<<endl;
cin>>publication;
```

```
cout<<"Enter pages: "<<endl;
cin>>pages;
}

void book::show()
{
cout<<"\nBook Details: ";
cout<<"\nTitle: "<<title;
cout<<"\nPublication: "<<publication;
cout<<"\nPages: "<<pages;
}

void tape::read()
{
cout<<"\nEnter title of tape: "<<endl;
cin>>title;
cout<<"Enter publication: "<<endl;
cin>>publication;
cout<<"Enter playing time (seconds): "<<endl;
cin>>time;
}

void tape::show()
{
cout<<"\nTape Details: ";
cout<<"\nTitle: "<<title;
cout<<"\nPublication: "<<publication;
cout<<"\nPlaying Time: "<<time;
}

int main()
{
media *p;
int i=1,choice;
book b;
tape t;
do
{
cout<<"\n\nEnter your choice: 1.Book 2.Tape 3.Exit ";
cin>>choice;
switch(choice)
{
case 1:
p=&b;
p->read();
p->show();
break;
case 2:
p=&t;
p->read();
p->show();
break;
case 3:
i=0;
break;
default:
cout<<"Invalid choice";
}
}
while(i);
return 0;
}
```

Output-



```
Enter title of book:
ABC
Enter publication:
WER
Enter pages:
33

Book Details:
Title: ABC
Publication: WER
Pages: 33

Enter your choice: 1.Book 2.Tape 3.Exit 2

Enter title of tape:
QWE
Enter publication:
RRR
Enter playing time (seconds):
55

Tape Details:
Title: QWE
Publication: RRR
Playing Time: 55

Enter your choice: 1.Book 2.Tape 3.Exit 3

--------------------------------
Process exited after 25.83 seconds with return value 0
Press any key to continue . . .
```

Conclusion- The concept of virtual function in C++ has been studied.

Remarks-

Signature-

PRACTICAL8

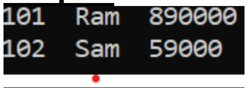
Aim-Write a program to show the use of **this pointer.**

Software Used-Dev C++

Program-

```
#include <iostream>
using namespace std;
class Employee
{
   public:
       int id;
       string name;
       float salary;
       Employee(int id, string name, float salary)
        {
            this->id = id;
           this->name = name;
           this->salary = salary;
        }
       void display()
        {
           cout<<id<<"  "<<name<<"  "<<salary<<endl;
        }
};
int main(void)
 {
    Employee e1 =Employee(101, "Ram", 890000);
    Employee e2=Employee(102, "Sam", 59000);
    e1.display();
    e2.display();
    return 0;
}
```

Output-



```
101  Ram  890000
102  Sam  59000
```

Conclusion- The concept of this pointer in C++ has been studied.

Remarks-

Signature-