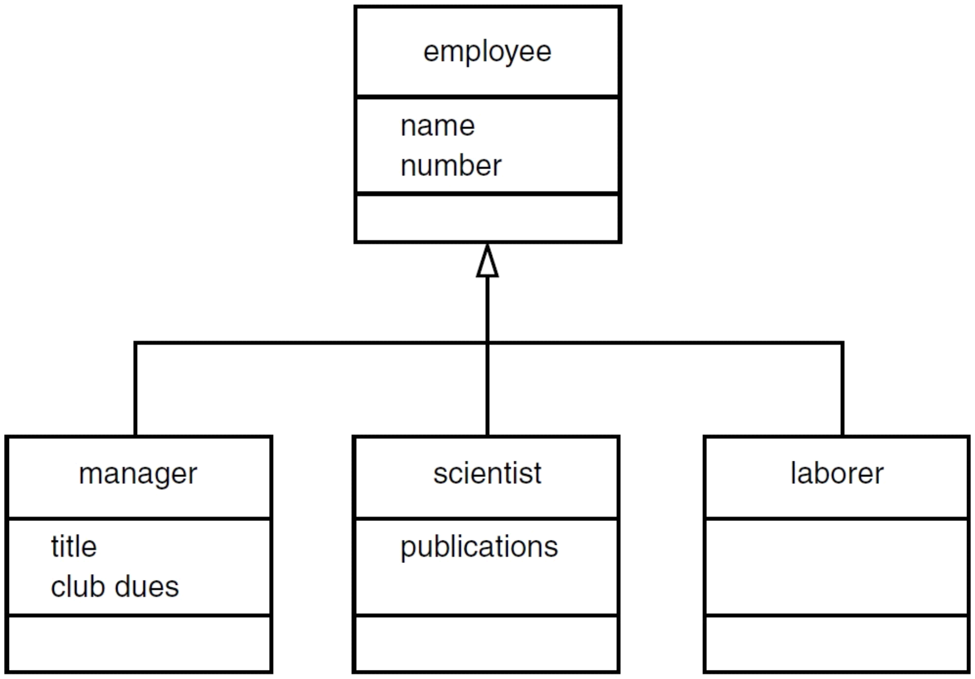
(SINGLE INHERITANCE)

(MADE BY ALI AKBER)

(BSCS 2ND SS1)

QNO1:



```
#include<iostream>

#include<string.h>


using namespace std;

class Person

{

    private:

        int id;

        char name[50];

    public:

        Person(): id(0)
```

```
            {
                    strcpy(name,"Null");
            }
        Person(int i, char na[]): id(i)
        {
                strcpy(name, na);
        }
        void showdata()
        {
                cout<<"Id is "<<id<<endl;
                cout<<"Name is "<<name<<endl;
        }
        void getdata()
        {
                cout<<"Enter id ";
                cin>>id;
                cout<<"Enter name ";
                cin>>name;
        }
};
class Student : public Person
{
        private:
                float gpa;
        public:
                Student(): Person(), gpa(0) {}
                Student(int i, char na[], float gp): Person(i, na), gpa(gp) {}
                void showdata()
        {
            Person::showdata();
            cout<<"Gpa "<<gpa<<endl;
        }
        void getdata()
        {
            Person::getdata();
            cout<<"Enter Gpa ";
            cin>>gpa;
        }
};
class Teacher : public Person
{
    private:
        float salary;
    public:
        Teacher(): Person(), salary(0.0) {}
        Teacher(int i, char na[], float sal): Person(i, na), salary(sal) {}
        void showdata()
        {
            Person::showdata();
            cout<<"Salary    "<<salary<<endl;
        }
        void getdata()
        {
            Person::getdata();
            cout<<"Enter Salary ";
            cin>>salary;
```

```
                }
};
int main()
{
        Student s1(1,"Ali", 3.41f);
        Teacher t1;

        s1.showdata();
        cout<<endl;

        t1.getdata();
        cout<<endl;
        t1.showdata();
            return 0;
}
```

<u>Q.2:</u> Create two classes:

Rectangle

The *Rectangle* class should have two data members-*width* and *height* of *float* types. The class should have *display ()* member function, to print the *width* and *height* of the rectangle separated by space.

RectangleArea

The *RectangleArea* class is derived from the *Rectangle* class, i.e., it is the sub-class of *Rectangle* class. The class should have *read_input()* member function, to read the values of *width* and *height* of the rectangle. The *RectangleArea* class should also override the *display()* member function to print the area (*width * height*) of the rectangle.

<u>SOLUTION</u>

```
#include <iostream>
#include <string.h>
using namespace std;
```

```
class Rectangle{
        protected:
        float height;
        float width;
        public:
                Rectangle():height(0),width(0){  }    //no arg constructor.
                Rectangle(float h,float w):height(h),width(w){     }    //two arg constructor.
                void showdata(){
                        cout<<"Height of rectangle is:"<<height<<endl;
                        cout<<"Width of rectangle is:"<<width<<endl;
                }
};
class Rectanglearea :public Rectangle {
public:
        Rectanglearea():Rectangle(){      }
                Rectanglearea(float w,float h):Rectangle(w,h){    }

void read_input(){
cout<<"Enter height of rectangle:"<<endl;
                        cin>>height;
                        cout<<"Enter width of rectangle:"<<endl;
                        cin>>width;
}
void display_input(){
        cout<<"Area is:"<<height*width<<endl;
}

};
```

```
int main (){
Rectanglearea r1;
r1.read_input();
r1.display_input();
        return 0;
}
```

Q.3: Imagine a publishing company that markets both book and audiocassette versions of its works. Create a class publication that stores the title (a string) and price (type float) of a publication. From this class derive two classes: book, which adds a page count (type int), and tape, which adds a playing time in minutes (type float). Each of these three classes should have a getdata() function to get its data from the user at the keyboard, and a putdata() function to display its data. Write a main () program to test the book and tape classes by creating instances of them, asking the user to fill in data with getdata(), and then displaying the data with putdata().

SOLUTION:

```
//SINGLE INHERITANCE
//PUBLICATION,BOOK AND TAPE QUESTION.
#include <iostream>
#include <string.h>
using namespace std;
class Publication{
private:
string title;
float price;
public:
        Publication():title(""),price(0){    }      //no arg constructor.
        Publication(string s,float p):title(s),price(p){        }      //two arg constructor.
void putdata(){
        cout<<"Enter the title of publication:"<<endl;
```

```
            cin>>title;
            cout<<"Enter the price of publication:"<<endl;
            cin>>price;
}
void getdata()const{
            cout<<"Title of the publication is:"<<title<<endl;
            cout<<"Price of the publication is:"<<price<<endl;
}
};
class Book :public Publication{
private:
int count;
public:
                        Book():Publication(){
                        count=0;            }       //no arg constructor.
            Book(string s,float p,int c):Publication(s,p){
            count=c;}       //three arg constructor.
            void putdata(){
                        Publication::putdata();
                        cout<<"Enter Book Pages:"<<endl;
                        cin>>count;
            }
            void getdata()const{
                        Publication::getdata();
                        cout<<"Book Pages is:"<<count<<endl;
            }
};
class Tape :public Publication{
```

```
private:
float minutes;
public:
        Tape():Publication(){
                minutes=0;      }     //no arg constructor.
        Tape(string s,float p,float m):Publication(s,p){
                minutes=m;}     //three arg constructor.
        void putdata(){
                Publication::putdata();
                cout<<"Enter Playing time of tape:"<<endl;
                cin>>minutes;
        }
        void getdata()const{
                Publication::getdata();
                cout<<"Playing time of tape is:"<<minutes<<endl;
        }
};
int main (){
Book b1;
b1.putdata();
b1.getdata();
cout<<endl;
Tape t1;
t1.putdata();
t1.getdata();
        return 0;
}
```

QNO4: (PAST PAPER QUESTION).

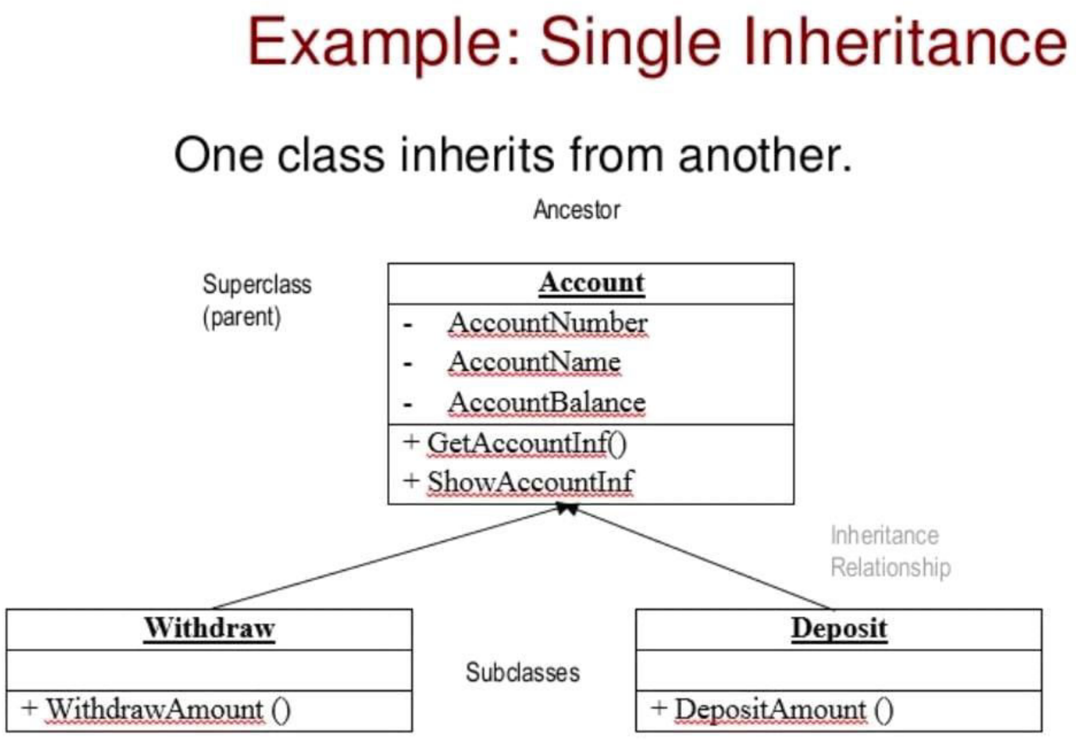
Write a program that create a class Course with two data members , course name "ename" with type c-string and course code "ccode" with type double.This class also include ls get() and show() member functions for input and output data members. From this class,create a derived class Lab. this class contains additional data member for lab credit hour "Lhour" with type integer. Derived class overrides the base class member functions.This class also overload the (==) operator for comparing two lab objects.Define appropriate constructors of the classes and makes appropriate member functions as const member functions. Create two objects of the Lab class in main, input the values, and compare the objects. If the course name "DLD" , "OOP"   and course code are 101 , 102 respectively with 04 credit hours then display a message in main() "Both lab courses are same", otherwise display a message "Both lab courses are not same".

```
#include <iostream>

#include <string.h>

using namespace std;

class Course{

        private:

                char cname[50];

                double ccode;

                public:

                        Course(){

                                strcpy(cname,"");

                                ccode=0;

                        }

                        Course(char na[],double cd ){

                                strcpy(cname,na);

                                ccode=cd;

                        }

                void getdata(){

                        cout<<"Enter name of the course:"<<endl;

                        cin.ignore();

                        cin.getline(cname,50);

                        cout<<"Enter the course code:"<<endl;

                        cin>>ccode;
```

```
                    }
                              void showdata()const{
                                        cout<<"Course name is:"<<cname<<endl;
                                        cout<<"Course code is:"<<ccode<<endl;
                              }
};
class Lab :public Course{
          private:
                    int Lhour;
                    public:
          Lab():Course(){
                              Lhour=0;            }
                              Lab(char na[],double cd,int h ):Course(na,cd){
                              Lhour=h;
                              }
                              void getdata(){
                                        Course::getdata();
                                        cout<<"Enter Lab Hours:"<<endl;
                                        cin>>Lhour;
                              }
                              void showdata()const{
                                        Course::showdata();
                                        cout<<"Lab Hours are :"<<Lhour<<endl;
                              }
                              bool operator ==(const Lab&l3){
                              if ( Lhour == l3.Lhour){
                                        return 1;
                              }
```

```
                                        else{
                                        return 0;
                                        }
                                        }
};
int main (){
Lab l1,l2;
cout << "Enter details for Lab 1" << endl;
l1.getdata();
cout<<endl;
cout << "Details of Lab 1" << endl;
l1.showdata();
cout<<endl;
cout << "Enter details for Lab 2" << endl;
l2.getdata();
cout<<endl;
cout << "Details of Lab 2" << endl;
l2.showdata();
cout<<endl;
cout << "Comparing Lab 1 and Lab 2" << endl;
if (l1 == l2) {
          cout << "Both Courses are same" << endl;
     }
     else {
          cout << "Both Courses are not same " << endl;
     }
        return 0;
}
```

QNO5:



```
#include <iostream>

#include <string.h>

using namespace std;

class Account{

        protected:

                int AccountNumber;

                string AccountName;

                float AccountBalance;

                public:

                        Account(){

                        AccountNumber=0;

                        AccountBalance=0;

                        AccountName="Null";
```

```
            }
                Account(int ano,string ana,float ab){
            AccountNumber=ano;
            AccountBalance=ab;
            AccountName=ana;
            }
            void GetAccountInf(){
                cout<<"Enter account number:"<<endl;
                cin>>AccountNumber;
                cout<<"Enter account name:"<<endl;
                cin.ignore();
                getline(cin,AccountName);
                cout<<"Enter account balance:"<<endl;
                cin>>AccountBalance;
            }
            void ShowAccountInf(){
                cout<<"Account Number is:"<<AccountNumber<<endl;
                cout<<"Account Name is:"<<AccountName<<endl;
                cout<<"Account Balance is:"<<AccountBalance<<endl;
            }
};
    class Withdraw :public Account{
        public:
            Withdraw():Account(){   }
            Withdraw(int ano,string ana,float ab):Account(ano,ana,ab){        }
            void WithdrawAmount(){
                double withdrawamount;
                cout<<"Enter amount to be withdrawn:"<<endl;
```

```
                    cin>>withdrawamount;
                    if (withdrawamount>AccountBalance){
                cout<<"Your balance is insufficient for this amount:"<<endl;
                    }
                    else {
                        AccountBalance -=withdrawamount;
                            cout<<"Amount Withdrawl Successfully!"<<endl;
                    }
                }
};
class Deposit :public Account{
        public:
                Deposit():Account(){      }
                Deposit(int ano,string ana,float ab):Account(ano,ana,ab){           }
                void DepositAmount(){

                    double Depositamount;
                    cout<<"Enter amount to be deposit:"<<endl;
                    cin>>Depositamount;
                    if (Depositamount<0){
                cout<<"Invalid amount entered:"<<endl;
                    }
                    else {
                        AccountBalance +=Depositamount;
                            cout<<"Amount Deposited Successfully!"<<endl;
                    }
                }
};
```

```
int main (){
            cout<<"Details of the account:"<<endl;
Account obj1;
        obj1.GetAccountInf();
        cout<<endl;
        obj1.ShowAccountInf();
        cout<<"Details of the account after withdrawl:"<<endl;
        Withdraw obj2;
        obj2.GetAccountInf();
        obj2.ShowAccountInf();
        obj2.WithdrawAmount();
        obj2.ShowAccountInf();
        cout<<"Details of the account after deposit:"<<endl;
        Deposit obj3;
        obj3.GetAccountInf();
        obj3.DepositAmount();
        obj3.ShowAccountInf();
            return 0;
}
```