**Programs:**

## Program to read and print employee information with department and loan information using hierarchical inheritance program in C++.

## Class BasicInfo: name, empID, gender

## Class DeptInfo: deptName, designation, salary

## Class LoanInfo: LoanType, Amount

#include<iostream>

#include<conio.h>

#include<string>

using namespace std;

class BasicInfo

{

string name, gender;

int empID;

public:

void getb()

{

cout << "Enter name,employee id and gender:\n";

cin >> name >> empID >> gender;

}

void displayb()

{

cout << "Employee name: " << name << endl;

cout << "Employee ID: " << empID << endl;

cout << "Employee gender: " << gender << endl;

}

};

class DeptInfo:public BasicInfo

{

string deptName, designation;

int salary;

public:

void getd()

{

BasicInfo::getb();

cout << "Enter department name,designation and salary:\n";

cin >> deptName >> designation >> salary;

}

void displayd()

{

BasicInfo::displayb();

cout << "Employee department name: " << deptName << endl;

cout << "Employee designation: " << designation << endl;

cout << "Employee salary: " << salary << endl;

}

};

class LoanInfo:public BasicInfo

{

string LoanType, Amount;

public:

void getl()

{

BasicInfo::getb();

cout << "Enter loan type and amount " << endl;

cin >> LoanType >> Amount;

}

void displayl()

{

BasicInfo::displayb();

cout << "Loan type: " << LoanType << endl;

cout << "Amount: " << Amount << endl;

}

};

int main()

{

DeptInfo d;

LoanInfo l;

d.getd();

d.displayd();

cout << endl;

l.getl();

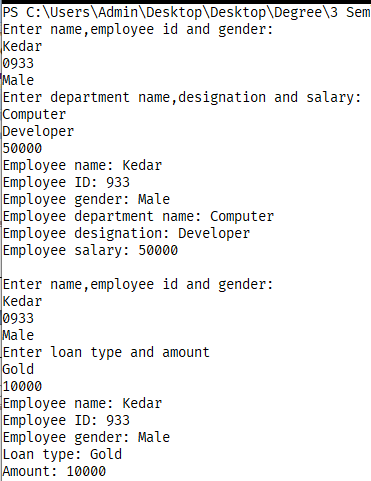
l.displayl();

\_getch();

return 0;

}

**Output:**

****

## Program to implement Hybrid inheritance

## Write a Program to design a student class representing student roll no. and a test class (derived class of student) representing the scores of the student in various subjects and sports class representing the score in sports. The sports and test class should be inherited by a result class having the functionality to add the scores and display the final result for a student

## #include<iostream>

## using namespace std;

## class student

## {

## int roll;

## public:

## void getinfo()

## {

## cout << "Enter Roll no: ";

## cin >> roll;

## cout << endl;

## }

## };

## class test :public student

## {

## int mrks[5];

## protected:

## double total=0;

## public:

## void gett()

## {

## cout << "Enter marks for 5 subjects:\n";

## for (int i = 0;i <= 4;i++)

## cin >> mrks[i];

## }

## void showt()

## {

## for (int i = 0;i <= 4;i++)

## total += mrks[i];

## }

## void displayt()

## {

## cout << "Entered marks are: \n";

## for (int i = 0;i <= 4;i++)

## cout<< mrks[i]<<" ";

## }

## };

## class sports

## {

## protected:int smrks;

## public:

## void gets()

## {

## cout <<"Enter sports marks:\n";

## cin >> smrks;

## }

## void displays()

## {

## cout << "\nEntered sports marks are: " << smrks << endl;

## }

## };

## class result:public sports,public test

## {

## float res;

## public:

## void totalr()

## {

## res = total + smrks;

## res = res / 6;

## }

## void displayr()

## {

## cout << "\nResult is: " << res << endl;

## }

## };

## int main()

## {

## result r;

## r.getinfo();

## r.gett();

## r.gets();

## cout << endl;

## r.showt();

## r.displayt();

## r.displays();

## r.totalr();

## r.displayr();

## return 0;

## }

## Output:

## 

## Program to initialize base class data members using constructors in derived class

## Class Person: name, age

## Class FootballPlayer: team, numberOfGoals

## #include<iostream>

## #include<conio.h>

## #include<string.h>

## #include<cstring>

## using namespace std;

## class person

## {

## protected :

## string name;

## int age;

## public:

## person(string s ,int a)

## {

## name = s;

## age = a;

## }

## };

## class FootballPlayer:public person

## {

## string team;

## int numberOfGoals;

## public:

## FootballPlayer(string s,int a,string t, int n) :person(s,a)

## {

## team = t;

## numberOfGoals = n;

## }

## void show()

## {

## cout << "Player name: " << name << endl;

## cout << "Player age: " << age << endl;

## cout << "Player team: " << team << endl;

## cout << "Number of goals: " << numberOfGoals << endl;

## }

## };

## int main()

## {

## char s[30], t[30];

## int a, n;

## cout << "Enter person's name, age, team and number of goals\n";

## cin >> s>> a>> t >> n;

## FootballPlayer p(s,a,t,n);

## p.show();

## return 0;

## }

## Output:

## 

## Program to show run time behaviour of virtual functions.

## A class shape having 2 ints as data members is inherited by class rectangle and class triangle. Write a function area that returns the area of triangle and rectangle as class members of Class Triangle and Rectangle respectively. Use the object of class shape to invoke the functions.

## #include<iostream>

## #include<conio.h>

## using namespace std;

## class shape

## {

## protected:

## int x, y;

## public:

## void get()

## {

## cout << "Enter value for X and Y\n";

## cin >> x >> y;

## }

## virtual void area()

## {

## }

## };

## class triangle:public shape

## {

## public:

## void area()

## {

## cout<<"Area of triangle is :"<<0.5\*x\*y;

## }

## };

## class rectangle:public shape

## {

## public:

## void area()

## {

## cout<<"Area of rectangle is :"<< x\*y;

## }

## };

## int main()

## {

## shape s;

## shape \*bptr;

## rectangle r;

## triangle t;

## r.get();

## bptr = &r;

## bptr->area();

## cout << endl<<endl;

## t.get();

## bptr = &t;

## bptr->area();

## return 0;

## }

## Output:

## 

## Consider a book shop which sells both books and video-tapes. Create a class known as media that stores the title publication. Create two derived classes, one storing the number of pages in a book and another for storing the playing time of a tape. Implement Run time polymorphism by using display() in the classes. Display() displays contents of tapes and books

## #include<iostream>

## #include<string>

## #include<cstring>

## #include<conio.h>

## using namespace std;

## class media

## {

## protected:

## char title[50];

## public:

## media(char\*s)

## {

## strcpy(title, s);

## }

## virtual void display()

## {}

## };

## class book :public media

## {

## int pages;

## public:

## book(char\*s, int n):media(s)

## {

## pages = n;

## }

## void display()

## {

## cout << "Tiltle: " << title;

## cout << "\nNumber of pages: " << pages;

## }

## };

## class video\_tape :public media

## {

## int time;

## public:

## video\_tape(char\*s, int t) :media(s)

## {

## time = t;

## }

## void display()

## {

## cout << "Tiltle: " << title;

## cout << "\nPlaying time of tape: " <<time;

## }

## };

## int main()

## {

## char\*title = new char[30];

## int n, t;

## cout << "The book details:\n";

## cout << "Enter book title: ";

## cin >> title;

## cout << "Enter number of pages: ";

## cin >> n;

## cout << endl;

## book b(title, n);

## cout << "The video tape details:\n";

## cout << "Enter video tape title: ";

## cin >> title;

## cout << "Enter playing time of tape: ";

## cin >> t;

## cout << endl;

## video\_tape v(title,t);

## media \*list[2];

## list[0] = &b;

## list[1] = &v;

## cout << "Book details: \n";

## list[0]->display();

## cout << "\n\nVideo tape details: \n";

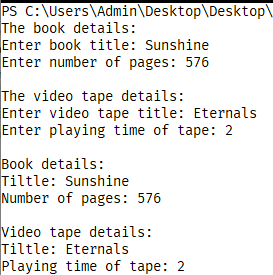
## list[1]->display();

## return 0;

## 

## }

**Output:**

****