Programs :

1. **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<stdlib.h>
#include<string>

using namespace std;
class BasicInfo
{
      string name, gender;
      int empID;
public:
      void getb()
      {
            cout << "Enter employee name: ";
            cin >> name;
            cout << "Enter employee ID: ";
            cin >> empID;
            cout << "Enter employee gender: ";
            cin >> 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: ";
            cin >> deptName;
            cout << "Enter designation: ";
            cin >> designation;
            cout << "Enter salary: ";
            cin >> 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: ";
		cin >> LoanType;
		cout << "Enter loan amount: ";
		cin >> Amount;
	}
	void displayl()
	{
		BasicInfo::displayb();
		cout << "Loan type: " << LoanType << endl;
		cout << "Loan Amount: " << Amount << endl;
	}
};
int main()
{
	DeptInfo d;
	LoanInfo l;

	cout << "----- Employee & Department Information -----" << endl;
	d.getd();
	cout << "\n----- Details -----" << endl;
	d.displayd();

	cout << endl;

	cout << "----- Employee & Loan Information -----" << endl;
	l.getl();
	cout << "\n ----- Details -----" << endl;
	l.displayl();

	_getch();
	return 0;
}
```

Output:

```
----- Employee & Department Information -----
Enter employee name: Rohan
Enter employee ID: 151
Enter employee gender: Male
Enter department name: Production
Enter designation: SDE1
Enter salary: 150000

----- Details -----
Employee name: Rohan
Employee ID: 151
Employee gender: Male
Employee department name: Production
Employee designation: SDE1
Employee salary: 150000

----- Employee & Loan Information -----
Enter employee name: Tanisha
Enter employee ID: 315
Enter employee gender: Female
Enter loan type: Gold
Enter loan amount: 130000

 ----- Details -----
Employee name: Tanisha
Employee ID: 315
Employee gender: Female
Loan type: Gold
Loan Amount: 130000
```

2. 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>
#include <conio.h>
using namespace std;
class student
{
      int rollNo;

public:
      void getinfo()
      {
            cout << "Enter Roll no: ";
            cin >> rollNo;
            cout << endl;
      }
};

class test : public student
{
      int marks[5];

protected:
      double total;

public:
      void gett()
      {
            cout << "Enter marks for the following subjects out of
25:\n";
            for (int i = 0; i <= 4; i++)
            {
                  cout << "CE" << i << " = ";
                  cin >> marks[i];
            }
      }
      void showt()
      {
            for (int i = 0; i <= 4; i++)
                  total += marks[i];
      }
      void displayt()
      {
            cout << "Entered marks are: \n";
            for (int i = 0; i <= 4; i++)
                  cout << marks[i] << " ";
      }
};

class sports
{
protected:
      int sMarks;
```

```
public:
	void gets()
	{
		cout << "Enter sports marks: ";
		cin >> sMarks;
	}
	void displays()
	{
		cout << "\nSports marks are: " << sMarks << endl;
	}
};

class result : public sports, public test
{
	float res;

public:
	void totalr()
	{
		res = total + sMarks;
		res = (res/ 150) * 100;
	}
	void displayr()
	{
		cout << "\nResult : " << res << "%" << endl;
	}
};

int main()
{
	result r;
	r.getinfo();
	r.gett();
	r.gets();
	cout << endl;
	r.showt();
	r.displayt();
	r.displays();
	r.totalr();
	r.displayr();
	_getch();
	return 0;
}
```

Output:

```
Enter Roll no: 1803

Enter marks for the following subjects out of 25:
CE0 = 23
CE1 = 21
CE2 = 20
CE3 = 22
CE4 = 24
Enter sports marks: 20

Entered marks are:
23 21 20 22 24
Sports marks are: 20

Result : 86.6667%
```

3. 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>

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 name[30], teamName[30];
	int age, goals;
	cout << "Enter player's name: ";
	cin >> name;
	cout << "Enter player's age: ";
	cin >> age;
	cout << "Enter player's team name: ";
	cin >> teamName;
	cout << "Enter the no of goals scored by the player: ";
	cin >> goals;
	FootballPlayer player1(name,age,teamName,goals);

	cout << "----- Details -----" << endl;
	player1.show();
	_getch();
	return 0;
}
```

Output:

```
Enter player's name: Messi
Enter player's age: 34
Enter player's team name: PSG
Enter the no of goals scored by the player: 756
----- Details -----
Player name: Messi
Player age: 34
Player team: PSG
Number of goals: 756
```

4. **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;
        cout << "----- Area of rectangle -----" << endl;
        r.get();
        bptr = &r;
        bptr->area();
        cout << endl << endl;
        cout << "----- Area of triangle -----" << endl;
        t.get();
        bptr = &t;
        bptr->area();
        _getch();
        return 0;
    }
```

Output:

```
----- Area of rectangle -----
Enter value for X and Y
3
4
Area of rectangle is :12

----- Area of triangle -----
Enter value for X and Y
5
6
Area of triangle is :15
```

5. 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<conio.h>
using namespace std;
class media
{
protected:
    char title[50];
    char publication[50];
public:
    media(char*titleN, char*publicationN)
    {
        strcpy(title, titleN);
        strcpy(publication, publicationN);
    }
    virtual void display()
    {}
};
```

```
class book :public media
{
    int pages;
public:
    book(char*titleN, char*publicationN, int noOfPages):media(titleN,
publicationN)
    {
          pages = noOfPages;
    }
    void display()
    {
          cout << "Tiltle: " << title << endl;
          cout << "Publication: " << publication << endl;
          cout << "Number of pages: " << pages << endl;
    }
};

class video_tape :public media
{
    int time;
public:
    video_tape(char*titleN, char*publicationN, int timeOftape)
:media(titleN, publicationN)
    {
          time = timeOftape;
    }
    void display()
    {
          cout << "Tiltle: " << title << endl;
          cout << "Publication: " << publication << endl;
          cout << "Playing time of tape: " << time << " Minutes." << endl;
    }
};

int main()
{
    char*title = new char[30];
    char*publication = new char[30];
    int noOfPages, timeOftape;
    cout << "----- Book Details -----" << endl;
    cout << "Enter book title: ";
    cin >> title;
    cout << "Enter book publication: ";
    cin >> publication;
    cout << "Enter number of pages: ";
    cin >> noOfPages;
    cout << endl;
    book b(title, publication, noOfPages);

    cout << "----- Video Tape Details -----" << endl;
    cout << "Enter video tape title: ";
    cin >> title;
    cout << "Enter book publication: ";
    cin >> publication;
    cout << "Enter playing time of tape: ";
    cin >> timeOftape;
    cout << endl;
    video_tape v(title, publication,timeOftape);

    media *list[2];
```

```
    list[0] = &b;
    list[1] = &v;
    cout << "----- Book Details -----" << endl;
    list[0]->display();
    cout << "\n\n----- Video Tape Details -----" << endl;
    list[1]->display();

    _getch();
    return 0;

}
```

Output: