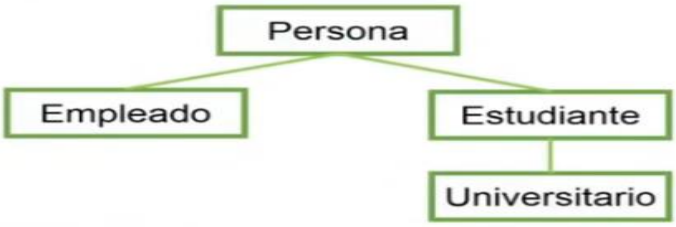
Herencia en POO

```cpp
/*HERENCIA EN POO*/

#include<iostream>
#include<stdlib.h>
using namespace std;

class Persona {
private: //Atributos
    string nombre;
    int edad;
public: //Metodos
    Persona(string, int); //Constructor Persona
    void mostrarPersona();
};

class Alumno: public Persona {
private: //Atributos
    string codigoAlumno;
    float notaFinal;
public: //Metodos
    Alumno(string, int, string,float); //Constructor Empleado
    void mostrarAlumno();
};
//Constructor de la clase persona
Persona::Persona(string _nombre, int _edad) {
    nombre = _nombre;
    edad = _edad;
}
//Constructor de la clase Alumno
Alumno::Alumno(string _nombre, int _edad, string _codigoAlumno, float _notaFinal)
    :Persona(_nombre, _edad) {
    codigoAlumno = _codigoAlumno;
    notaFinal = _notaFinal;
}
//Metodo Persona
void Persona::mostrarPersona() {
    cout << " Nombre: " << nombre << endl;
    cout << "Edad: " << edad << endl;
}
//Metodo Alumno
void Alumno::mostrarAlumno() {
    mostrarPersona();
    cout << "Codigo Alumno: " << codigoAlumno << endl;
    cout << "Nota Final: " << notaFinal << endl;
}

int main() {
    Alumno alumno1("Alberto", 20, "3350-12", 89.7);
```

```
49
50      alumno1.mostrarAlumno();
51
52      system("pause");
53      return 0;
54  }
55
```

Ejercicio : Realice un programa en C++, de tal manera que se construya una solución para la jerarquía(herencia) de clases mostrada en la siguiente figura.




```
1   /*Realice un programa en C++, de tal manera que se construya una
2   solución para la jerarquía(herencia) de clases mostrada en la siguiente figura
3   */
4
5   #include<iostream>
6   #include<stdlib.h>
7   using namespace std;
8
9   class Persona {
10  private: //Atributos
11      string nombre;
12      int edad;
13  public: //Metodos
14      Persona(string, int); //Constructor Persona
15      void mostrarPersona();
16  };
17
18  class Empleado : public Persona {
19  private: //Atributos
20      float sueldo;
21  public: //Metodos
```

```
    Empleado(string, int, float); //Constructor Empleado
    void mostrarEmpleado();
};

class Estudiante : public Persona {
private: //Atributos
    float notaFinal;
public: //Metodos
    Estudiante(string, int, float);
    void mostrarEstudiante();
};

class Universitario : public Estudiante {
private: //Atributos
    string carrera;
public:
    Universitario(string, int, float, string); //Constructor Universitario
    void mostrarUniversitario();
};

//Constructor de la clase Persona(Clase Padre)
Persona::Persona(string _nombre, int _edad) {
    nombre = _nombre;
    edad = _edad;
}

void Persona::mostrarPersona() {
    cout << "Nombre: " << nombre << endl;
    cout << "Edad: " << edad << endl;
}

//Constructor de la clase Empleado (Clase Hija)
Empleado::Empleado(string _nombre, int _edad, float _sueldo) : Persona(_nombre, _edad) {
    sueldo = _sueldo;
}

void Empleado::mostrarEmpleado() {
    mostrarPersona();
    cout << "Sueldo: " << sueldo << endl;
}
```

```
62
63  //Constructor de la clase Estudiante
64  Estudiante::Estudiante(string _nombre, int _edad, float _notaFinal) : Persona(_nombre, _edad) {
65      notaFinal = _notaFinal;
66  }
67
68  void Estudiante::mostrarEstudiante() {
69      mostrarPersona();
70      cout << "Nota Final: " << notaFinal << endl;
71  }
72
73  //Constructor de la clase Universitario(Clase Hija)
74  Universitario::Universitario(string _nombre, int _edad, float _notaFinal, string _carrera)
75      : Estudiante(_nombre, _edad, _notaFinal) {
76      carrera = _carrera;
77  }
78
79  void Universitario::mostrarUniversitario() {
80      mostrarEstudiante();
81      cout << "Carrera: " << carrera << endl;
82  }
83
84  int main() {
85      Empleado empleado1("Juan", 35, 1300);
86      cout << "-Empleado-" << endl;
87      empleado1.mostrarEmpleado();
88      cout << "\n";
89
90      Estudiante estudiante1("Maria", 21, 16.7);
91      cout << "-Estudiante-" << endl;
92      estudiante1.mostrarEstudiante();
93      cout << "\n";
94
95      Universitario universitario1("Alejandro", 20, 15.6, "Informatica");
96      cout << "-Universitario-" << endl;
97      universitario1.mostrarUniversitario();
98      cout << "\n";
99
100     system("pause");
101     return 0;
102 }
103
```