```
// inheritance1.h
#include <string>
using namespace std;

enum Discipline { ARCHEOLOGY, BIOLOGY, COMPUTER_SCIENCE };
enum Classification { Freshman, Sophomore, Junior, Senior };

class Person {
    protected:
        string name;
    public:
        Person() { setName("");}
        Person(string pName) { setName(pName);}
        void setName(string pName) { name = pName; }
        string getName() const { return name; }
};

class Student:public Person{
    private:
        Discipline major;
        Person *advisor;
    public:
        // Constructor
        Student(string sname, Discipline d, Person *adv);

        void setMajor(Discipline d) { major = d; }
        Discipline getMajor() const {return major; }
        void setAdvisor(Person *p){advisor = p;}
        Person *getAdvisor() const { return advisor; }
};

class Faculty:public Person{
    private:
        Discipline department;
    public:
        // Constructor
        Faculty(string fname, Discipline d) {
            // Access the protected base class member
            name = fname;
            department = d;
        }
        // Other member functions
        void setDepartment(Discipline d) { department = d; }
        Discipline getDepartment( ) const{ return department; }
};
```

```
// inheritance1.cpp
#include "inheritance1.h"
//*********************************************
// Constructor for the Student class.         *
//*********************************************
Student::Student(string sname, Discipline d, Person *adv) {
    // Access the  protected member name
    name = sname;

    // Access the other members
    major = d;
    advisor = adv;
}
```

```
// pr11-19.cpp
//This program demonstrates the use of objects of derived classes.

#include "inheritance1.h"
#include <iostream>
using namespace std;

// These arrays of string are used to print
// values of enumerated types
const string dName[] =
       {"Archeology", "Biology", "Computer Science" };

const string cName[] =
       {"Freshman", "Sophomore", "Junior", "Senior" };

int main() {
   // Create Faculty and Student objects
   Faculty prof("Indiana Jones", ARCHEOLOGY);
   Student st("Sean Bolster", ARCHEOLOGY, &prof);
   cout << "Professor " << prof.getName() << " teaches "
        << dName[prof.getDepartment()] << "." << endl;

   //Get student's advisor
   Person *pAdvisor = st.getAdvisor();
   cout << st.getName() <<"\'s advisor is "
        << pAdvisor->getName() << ".";

   return 0;
}
```

OUTPUT:

```
Professor Indiana Jones teaches Archeology.
Sean Bolster's advisor is Indiana Jones.
```