```
// inheritance4.h
#include <string>
#include <memory>
using namespace std;

enum Discipline { ARCHEOLOGY, BIOLOGY, COMPUTER_SCIENCE };
enum Classification { FRESHMAN, SOPHOMORE, JUNIOR, SENIOR };

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


// Student "is-a" Person
class Student:public Person {
    private:
        Discipline major;
        shared_ptr<Person> advisor;
    public:
        Student(const string& sname, Discipline d,
            const shared_ptr<Person>& adv) : Person(sname) {
            major = d;
            advisor = adv;
        }

        void setMajor(Discipline d) { major = d; }
        Discipline getMajor() const { return major; }
        void setAdvisor(shared_ptr<Person>& p) { advisor = p; }
        shared_ptr<Person> getAdvisor() const { return advisor; }
};


// Faculty "is-a" Person
class Faculty:public Person {
    private:
        Discipline department;
    public:
        Faculty(const string& fname, Discipline d) : Person(fname) {
            department = d;
        }
        void setDepartment(Discipline d) { department = d; }
        Discipline getDepartment( ) const { return department; }
};


// TFaculty "is-a" Faculty, which "is-a" Person
class TFaculty: public Faculty {
    private:
        string title;
    public:
        // This Constructor allows the specification of a title
        TFaculty(const string& fname, Discipline d, string title)
                  : Faculty(fname, d) {
            setTitle(title);
        }
```

```
        void setTitle(const string& title) { this->title = title; }

        // Override Person's getName function
        string getName( ) const{ return title + " " + name; }
};



// This exhibits the default non-polymorphic behavior of C++.
#include "inheritance4.h"
#include <vector>
#include <iostream>
using namespace std;

int main() {
     // Create a vector of pointers to Person objects
     vector<shared_ptr<Person>> people {
        make_shared<TFaculty>("Indiana Jones", Discipline::ARCHEOLOGY, "Dr."),
        make_shared<Student>("Thomas Cruise", Discipline::COMPUTER_SCIENCE, nullptr),
        make_shared<Faculty>("James Stock", Discipline::BIOLOGY),
        make_shared<TFaculty>("Sharon Rock", Discipline::BIOLOGY, "Professor"),
        make_shared<TFaculty>("Nicole Eweman", Discipline::ARCHEOLOGY, "Dr.")
     };

     // Print the names of the Person objects
     for (int k = 0; k < people.size(); k++) {
        cout << people[k]->getName() << endl;
     }

     return 0;
}
```