// inheritance3.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:

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:

Faculty(string fname, Discipline d) : Person(fname) {

department = d;

}

void setDepartment(Discipline d) { department = d; }

Discipline getDepartment( ) const { return department; }

};

class TFaculty: public Faculty {

private:

string title;

public:

// This Constructor allows the specification of a title

TFaculty(string fname, Discipline d, string title)

: Faculty(fname, d) {

setTitle(title);

}

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

// Override the getName function

string getName( ) const{ return title + " " + name; }

};

// inheritance3.cpp

#include "inheritance3.h"

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Constructor for the Student class. \*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Student::Student(string sname, Discipline d, Person \*adv)

: Person(sname) {

major = d;

advisor = adv;

}

// This program illustrates member function overriding.

#include "inheritance3.h"

#include <iostream>

using namespace std;

// These arrays of string are used to output

// values of enumerated types.

const string dName[] =

{ "Archeology", "Biology", "Computer Science" };

const string cName[] =

{ "Freshman", "Sophomore", "Junior", "Senior" };

int main() {

// New constructor allows specification of title

TFaculty prof("Indiana Jones", ARCHEOLOGY, "Dr.");

Student st("Sean Bolster", ARCHEOLOGY, &prof);

// Use the new TFaculty version of getName

cout << prof.getName() << " teaches "

<< dName[prof.getDepartment()] << "." << endl;

// This call uses the Person version of getName

Person \*pAdvisor = st.getAdvisor();

cout << st.getName() <<"\'s advisor is "

<< pAdvisor->getName() << ".";

return 0;

}

**OUTPUT:**