

Chapter 13 – Using Pre-existing Classes

One of the most frequently performed tasks of a C++ programmer is to use a user-created class within a program. While students spend most of their time learning how to create original code, professional programmers divide their time between writing original code and making use of classes that were previously created by other programmers.

Most programmers find it easier to use classes that they have created than to use classes that someone else has created. This chapter is an example with exercises in using pre-existing classes defined by someone else. The disk or compressed file contained this text has files *student2.h*, *course2.h* and *role2.h*, which will be used in this chapter. (If for some reason, this is not the case, you can obtain these files by email from fdd@louisiana.edu. They are also listed at the end of the chapter.) These files are written so that they are lacking only a main to be used. In consideration of beginning programming students, the class definition and member function implementations are all in one file per class.

The *class student* is located in the file *student2.h*. To use the *class student*, all that a programmer needs to look at is the class definition:

```
class student {
public:
    student( ;
    student(char * initName, int initId);
    char * getName( );
    int getId( );
private:
    char name[25];
    int id;
};
```

The *constructors* are used when declaring an object of *class student*. The function *getName* is used to retrieve the name stored in an object. The function *getId* is used to retrieve the value of *id* stored in an object. Here is an example program to use *class student*:

```

#include <iostream>
#include "student2.h"      / the full path may be needed here
using namespace std;

void main() {
    student student1("James Morrison", 23);
    student student2("Ringo Star", 99)
    cout << student1.getName() << " " << student1.getId() << endl;
    cout << student2.getName() << " " << student2.getId() << endl;
}

```

When run, this program will produce the following output:

```

James Morrison 23
Ringo Star 99

```

Programming Assignment 13.1

Add a third student to the example program at the top of page 2.

Programming Assignment 13.2

The *class course*, located in the file *course2.h*, has the following definition:

```

class course {
public:
    course();
    course(char * initCourseName, int initCourseId);
    char * getCourseName();
    int getCourseId();
private:
    char courseName[25];
    int courseId;
};

```

Write and run a C++ program that contains three objects of *class course*. The program should output all values in *courseName* and *courseId*.

Programming Assignment 13.3

The *class role*, located in the file *role2.h*, uses the *classes student* and *course* to create a class that will keep the role and grades of a course. *Class role* has the following definition:

```
class role {
public:
    role( );
    role(char * initCourseName, int initCourseId);
    void addStudent(char * studentName, int studentId);
    void setGrade(int studentId, char grade);
    char * getStudent(int studentId);
    char getGrade(int studentId);
    void listRole( );
private:
    course theClass;
    int numStudents;
    student courseRole[20];
    char courseGrade[20];
};
```

Create a program that uses an object of *class role*. Use the *member functions* of *class role* to create the course “English” with a course id of 101. The course should have at least five students. In addition, set the grades of each student. Lastly, output the course role.

Listing of *student2.h*

```
#ifndef _STUDENT
#define _STUDENT
#include <iostream>
#include <string.h>
using namespace std;

class student {
public:
    student();
    student(char * initName, int initId);
    char * getName();
    int getId();
public:
    char name[25];
    int id;
};

student::student() {
    strcpy(name, "(Nobody)");
    id = -1;
}

student::student(char * initName, int initId) {
    strcpy(name, initName);
    id = initId;
}

char * student::getName() {
    return name;
}

int student::getId() {
    return id;
}

#endif
```

Listing of *course2.h*

```
#ifndef _COURSE
#define _COURSE
#include <string.h>
using namespace std;

class course {
public:
    course();
    course(char * initCourseName, int initCourseId);
    char * getCourseName();
    int getCourseId();
public:
    char courseName[25];
    int courseId;
};

course::course( ) {
    strcpy(courseName,"!!nothing!!");
    courseId = -1;
}

course::course(char * initCourseName, int initCourseId) {
    strcpy(courseName,initCourseName);
    courseId = initCourseId;
}

char * course::getCourseName( ) {
    return courseName;
}

int course::getCourseId( ) {
    return courseId;
}

#endif
```

Listing of *role2.h*

```

#ifndef _ROLE
#define _ROLE
#include <iostream>
#include "student2.h"
#include "course2.h"
using namespace std;

const int maxstu=20;

class role {
public:
    role();
    role(char * initCourseName, int initCourseId);
    void addStudent(char * studentName, int studentId);
    void setGrade(int studentId, char grade);
    char * getStudent(int studentId);
    char getGrade(int studentId);
    void listRole();
private:
    course theClass;
    int numStudents;
    student courseRole[20];
    char courseGrade[20];
};

role::role() {
    theClass = course("-", -1);
    numStudents = 0;
}

role::role(char * initCourseName, int initCourseId) {
    theClass = course(initCourseName, initCourseId);
    numStudents = 0;
}

void role::addStudent(char * studentName, int studentId) {
    if (numStudents < maxstu)
    {
        courseRole[numStudents] = student(studentName, studentId);
        courseGrade[numStudents] = '-';
        numStudents++;
    }
}

```

```

void role::setGrade(int studentId, char grade) {
    for (int x=0; x < numStudents; x++)
        if (courseRole[x].getId() == studentId) break;
    if (x < numStudents)
        courseGrade[x] = grade;
}

char * role::getStudent(int studentId) {
    for (int x=0; x < numStudents; x++)
        if (courseRole[x].getId() == studentId) break;
    if (x < numStudents)
        return courseRole[x].getName();
    else
        return "-";
}

char role::getGrade(int studentId) {
    for (int x=0; x < numStudents; x++)
        if (courseRole[x].getId() == studentId) break;
    if (x < numStudents)
        return courseGrade[x];
    else
        return '-';
}

void role::listRole( ) {
    cout << endl << endl
        << "Class Name: " << theClass.getCourseName() << endl
        << "Class Id: " << theClass.getCourseId() << endl
        << endl << endl
        << "ID    Name          Grade" << endl
        << "===== " << endl;
    for (int x=0; x < numStudents; x++)
    {
        cout.width(3);
        cout << courseRole[x].getId();
        cout << " ";
        cout << courseRole[x].getName();
        cout.width( 20-strlen(courseRole[x].getName() ) );
        cout << ' ';
        cout << courseGrade[x] << endl;
    }
}

#endif

```