Polytechnic University of Puerto Rico
Electrical and Computer Engineering & Computer Science Department
CECS 2223 – Computer Programming II Lab

Lab 6

Name: Coral S. Schmidt Montilla ID#: 148830
Name: Aleysha M. Rivera Cores ID#: 108408

1. Copy the source code developed for Lab 5 and paste it as **text** below. (*15 points*)

```
/*
* CECS 2223, Computer Programming II Laboratory
* Fall 2023
* Date: October 10, 2023
* Topic: Lab 6 - Composition and Dynamic Memory
* File name: Catalog.h
* This file declares a class named Catalog
* Complete the declaration as required.
* Name: Coral S. Schmidt Montilla, ID#148830
* Name: Aleysha M. Rivera Cores, ID#108408
*/
#pragma once
// preprocessor directives
#include "Course.h"
class Catalog {
private:
	string universityName; // the name of the university
	int courseCount; // the number of courses in the catalog
	Course** courses; // the array which contains the courses
	void sortCatalog(); // sorts the courses in the catalog in alphabetical order

	friend bool execute(Catalog&, const int);
public:
	Catalog(); // the default constructor
	Catalog(const Catalog&); // the copy constructor
	~Catalog(); // the destructor

	void setUniversityName(string);
	string getUniversityName() const;
	void addCourse(string, string, int); // adds a course to the catalog
	// The addCourse method calls the sortCatalog method after a course is added.
	int findCourse(const string) const; // method to find a course in the catalog
	void deleteCourse(int); // removes a course from the catalog
	void printCatalog() const; // prints the course list for the university
	// printCatalog prints the header CODE, NAME, CREDITS using the column widths
	// from the Course class
	void operator=(const Catalog&); // overload of the assignment operator
	int getCount() const;
};
/*
* CECS 2223, Computer Programming II Laboratory
* Fall 2023
* Date: October 10, 2023
```

```
* Topic: Lab 6 - Composition and Dynamic Memory
* File name: Course.h
* This file declares a class named Course
* Complete the declaration as required.
* Name: Aleysha M. Rivera Cores, ID# 108408
* Name: Coral S. Schmidt Montilla, ID# 148830
*/
#pragma once
// preprocessor directives
#include <string>
#include <iostream>
using namespace std;
class Course {
private:
	string courseCode; // the course's code, e.g. CECS 2223
	string courseName; // the course's name, e.g. Computer Programming II
Laboratory
	int credits; // the number of credits for the course

	static size_t codeSize; // the size of the courseCode field
	void setCodeSize(string, bool); // private method to set the value for
codeSize
	static size_t nameSize; // the size of the courseName field
	void setNameSize(string, bool); // private method to set the value for
nameSize
public:
	Course(); // the default constructor
	Course(string, string, int); // parameterized constructor
	// the parameterized constructor must validate the value for credits
	Course(const Course&); // the copy constructor
	~Course(); // the destructor

	void setCourseCode(string);
	void setCourseName(string);
	void setCourseCredits(int); // validates the parameter to be 0 or greater

	string getCourseCode() const;
	string getCourseName() const;

	int getCourseCredits() const;

	size_t getCodeSize() const;
	size_t getNameSize() const;

	void displayCourse() const; // prints the course data ready for a table
	// the order is course code, course name, credits, and the size of the
	// course and name fields is variable
};
/*
* CECS 2223, Computer Programming II Laboratory
* Fall 2023
* Date: Ovtober 10, 2023
* Topic: Lab 6 - Composition and Dynamic Memory
* File name: Catalog.cpp
```

```
* This file declares a class named Catalog
* Complete the declaration as required.
* Name: Aleysha M. Rivera Cores, ID# 108408
* Name: Coral S. Schmidt Montilla, ID# 148830
*/
#include "Catalog.h"
Catalog::Catalog() {
	universityName = "";
	courseCount = 0;
	courses = nullptr;
}
Catalog::Catalog(const Catalog& other) {
	universityName = other.universityName;
	courseCount = other.courseCount;
	// Allocate memory for the courses array
	courses = new Course * [courseCount];
	// Deep copy the courses
	for (int i = 0; i < courseCount; i++) {
		courses[i] = new Course(*(other.courses[i]));
	}
}
Catalog::~Catalog() {
	for (int i = 0; i < courseCount; i++) {
		delete courses[i]; // Delete each Course object
	}
	delete[] courses; // Delete the array of Course pointers
	// No need to delete universityName;
// it's a std::string and will be managed automatically
}
void Catalog::sortCatalog() {
	for (int i = 0; i < courseCount - 1; i++) {
		int minIndex = i;
		for (int j = i + 1; j < getCount(); j++) {
			if (courses[j]->getCourseCode() < courses[minIndex]-
>getCourseCode()) {
				minIndex = j;
			}
		}
		if (minIndex != i) {
			Course* temp = courses[i];
			courses[i] = courses[minIndex];
			courses[minIndex] = temp;
		}
	}
}
void Catalog::setUniversityName(string name) {
	universityName = name;
}
string Catalog::getUniversityName() const {
	return universityName;
}
void Catalog::addCourse(string code, string name, int credits) {
	// Check if a course with the same code already exists
	int existingIndex = findCourse(code);
```

```
	if (existingIndex != -1) {
		cout << "Course with code " << code << " already exists in the catalog.
Duplicate course not added." << endl << endl;
	}
	else {
		// Create a new course
		Course* newCourse = new Course(code, name, credits);
		// Resize the array to accommodate the new course
		Course** newCourses = new Course * [courseCount + 1];
		for (int i = 0; i < courseCount; i++) {
			newCourses[i] = courses[i];
		}
		// Add the new course and update the course count
		newCourses[courseCount] = newCourse;
		delete[] courses;
		courses = newCourses;
		courseCount++;
		cout << "Course with code " << code << " added to the catalog." << endl
			<<
			endl;
		sortCatalog(); // Sort the catalog after adding
	}
}
int Catalog::findCourse(const string code) const {
	if (getCount() > 0) {
		for (int i = 0; i < getCount(); i++)
			if (courses[i]->getCourseCode().compare(code) == 0)
				return i;
	}
	return -1;
}
void Catalog::deleteCourse(int index) {
	if (index >= 0 && index < courseCount) {
		// Delete the course at the given index
		delete courses[index];
		// Create a new array without the deleted course
		Course** newCourses = new Course * [courseCount - 1];
		for (int i = 0, j = 0; i < courseCount; i++) {
			if (i != index) {
				newCourses[j++] = courses[i];
			}
		}
		delete[] courses;
		courses = newCourses;
		courseCount--;
	}
}
void Catalog::printCatalog() const {
	// Calculate the maximum lengths of course code and course name
	size_t maxCodeLength = 11; // "CODE" has 11 characters
	size_t maxNameLength = 11; // "NAME" has 11 characters
	for (int i = 0; i < courseCount; i++) {
		size_t codeLength = courses[i]->getCourseCode().length();
		size_t nameLength = courses[i]->getCourseName().length();
```

```
            if (codeLength > maxCodeLength) {
                  maxCodeLength = codeLength;
            }
            if (nameLength > maxNameLength) {
                  maxNameLength = nameLength;
            }
      }
      cout << "University: " << universityName << endl;
      cout << endl;
      printf("%-*s %--*s %s\n", int(maxCodeLength), "CODE", int(maxNameLength),
            "NAME", "CREDITS");
      cout << endl;
      for (int i = 0; i < courseCount; i++) {
            printf("%-*s %--*s %5d\n", int(maxCodeLength),
            courses[i]->getCourseCode().c_str(), int(maxNameLength), courses[i]-
>getCourseName().c_str(), courses[i] ->
            getCourseCredits());
            cout << endl;
      }
}
void Catalog::operator=(const Catalog& other) {
      if (this == &other) {
            return; // Avoid self-assignment
      }
      // Copy data from the other catalog
      universityName = other.universityName;
      courseCount = other.courseCount;
      courses = new Course * [courseCount];
      for (int i = 0; i < courseCount; i++) {
            courses[i] = new Course(*(other.courses[i]));
      }
}
int Catalog::getCount() const
{
      return courseCount;
}
/*
* CECS 2223, Computer Programming II Laboratory
* Fall 2023
* Date: October 10, 2023
* Topic: Lab 6 - Composition and Dynamic Memory
* File name: Course.cpp
* This file declares a class named Catalog
* Complete the declaration as required.
* Name: Aleysha M. Rivera Cores, ID# 108408
* Name: Coral S. Schmidt Montilla, ID# 148830
*/
#include "Course.h"
#include <iostream>
#include <string>
#include <iomanip>
using namespace::std;

size_t Course::codeSize = 5;
```

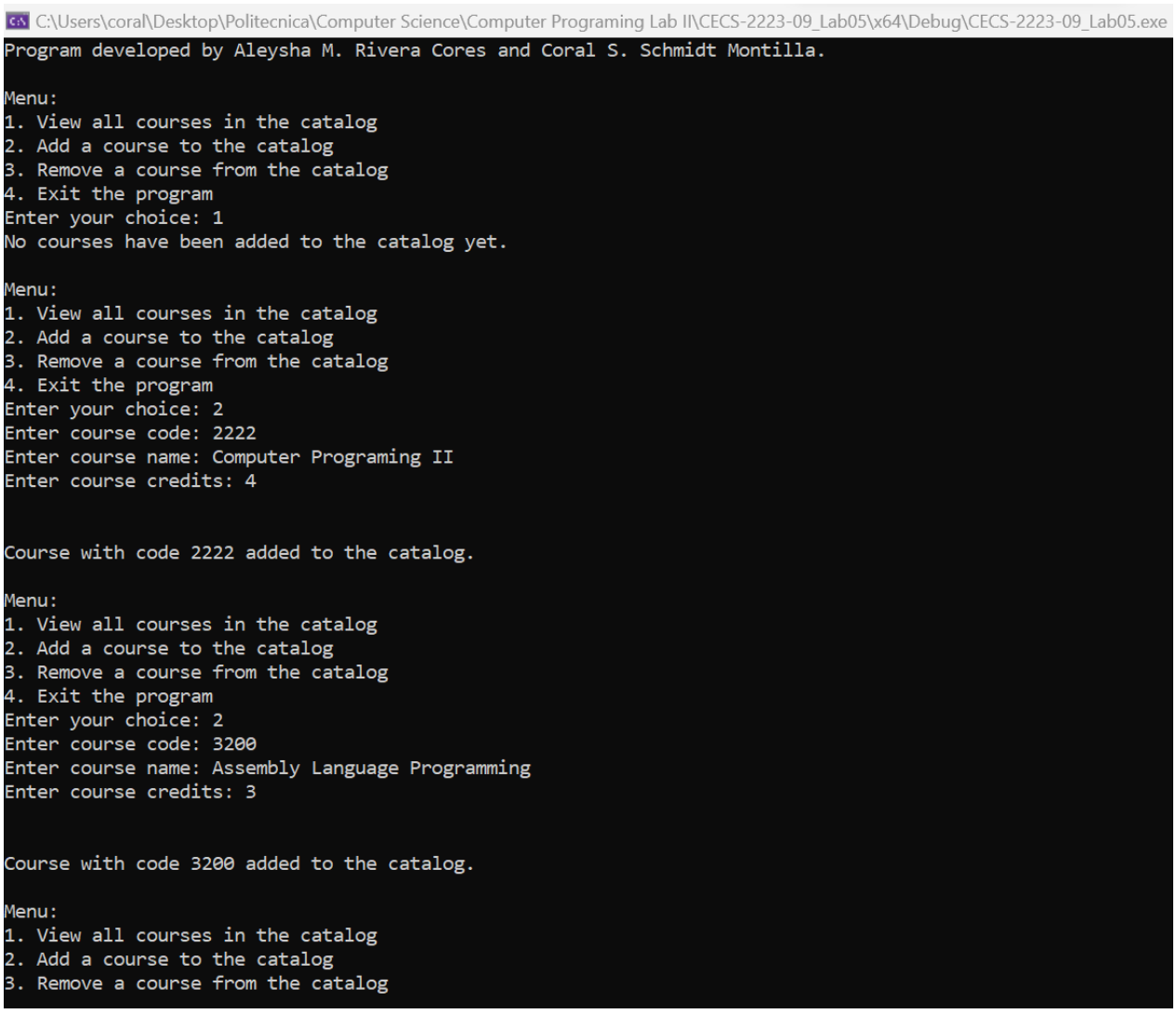
```
size_t Course::nameSize = 5;
//set codeSize based on courseCode

void Course::setCodeSize(string aCode, bool reduce) {
	if (reduce) {
		codeSize = aCode.size() + 1 > 5 ? aCode.size() + 1 : 5;
	}
	else {
		if (aCode.size() + 1 > getCodeSize())
			codeSize = aCode.size() + 1;
	}
}
//nameSize based on courseName
void Course::setNameSize(string aName, bool reduce) {
	if (reduce) {
		nameSize = aName.size() + 1 > 5 ? aName.size() + 1 : 5;
	}
	else {
		if (aName.size() + 1 > getNameSize())
			nameSize = aName.size() + 1;
	}
}
//d.constructor
Course::Course() {
	courseCode = "";
	courseName = "";
	credits = -1;
}
//p.constructor
Course::Course(string code, string name, int credits) {
	courseCode = code;
	setCodeSize(code, false);
	courseName = name;
	setNameSize(name, false);
	this->credits = credits > -1 ? credits : -1;
}
//copy constructor
Course::Course(const Course& aCourse) {
	courseCode = aCourse.getCourseCode();
	courseName = aCourse.getCourseName();
	credits = aCourse.getCourseCredits();
}
//destructor
Course::~Course() {
	cout << "\n\tThe Course with code " << getCourseCode() << " has been
eliminated.\n\n";
}
//Setter for crouseCode
void Course::setCourseCode(string code) {
	courseCode = code;
}
//Setter for courseName
void Course::setCourseName(string name) {
	courseName = name;
```

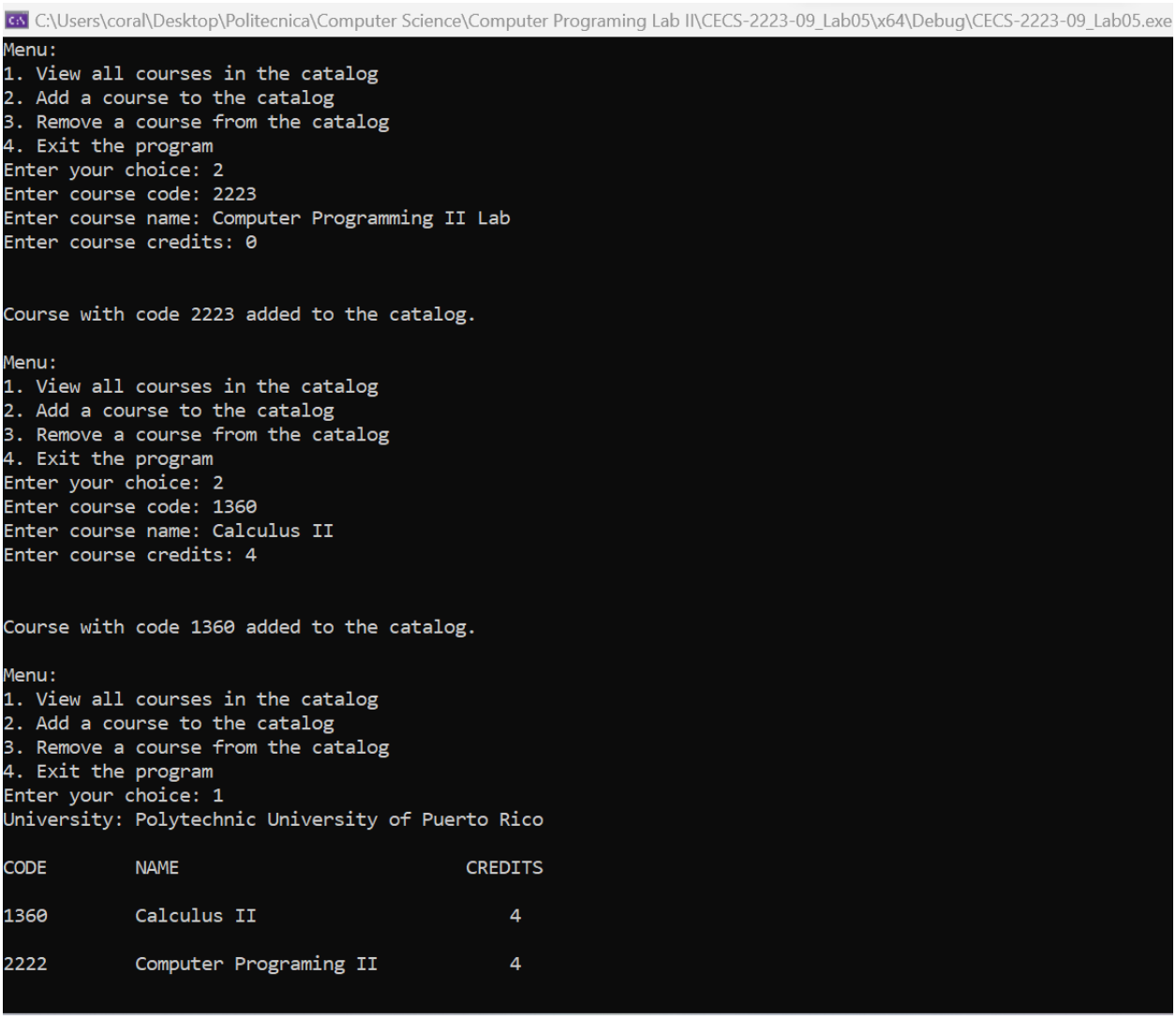
```
}
//setter for courseCredits
void Course::setCourseCredits(int c) {
	if (c >= 0) {
		credits = c;
	}
	else {
		cout << "Invalid credits value. Setting to 0." << endl;
		credits = 0;
	}
}
//Getter for courseCode
string Course::getCourseCode() const {
	return courseCode;
}
//Getter for courseName
string Course::getCourseName() const {
	return courseName;
}
//Getter for courseCredits
int Course::getCourseCredits() const {
	return credits;
}
//Getter for codeSize
size_t Course::getCodeSize() const {
	return codeSize;
}
//Getter for nameSize
size_t Course::getNameSize() const {
	return nameSize;
}
//Display course info
void Course::displayCourse() const {
	cout << left << setw(getCodeSize()) << getCourseCode() << setw(getNameSize())
<< getCourseName() << getCourseCredits() << endl;
}
/*
* CECS 2223, Computer Programming II Laboratory
* Fall 2023
* Date: September 27, 2023
* Topic: Lab 6 - Composition and Dynamic Memory
* File name: Lab06.cpp
* This file implements a class named Catalog
* Complete the code as required.
* Name: Aleysha M. Rivera Cores, ID# 108408
* Name: Coral S. Schmidt Montilla, ID# 148830
*
* To test your code, add at least 4 courses to the catalog, and remove at least 2.
* Start by selecting the option to print the catalog, then proceed to add courses,
* and finally remove courses. You should print the course list after every add or
* remove action.
*/
// preprocessor directives
#include "Catalog.h"
```

```
#include <iostream>
#include <string>
#include <iomanip>
using namespace std;
// Initialize static variables
int menu();
bool execute(Catalog&, const int);
void personalInfo();
string code, name;
int credits = 0;
int main() {
	Catalog poli;
	poli.setUniversityName("Polytechnic University of Puerto Rico");
	personalInfo();
	int option;
	do {
		option = menu();
	} while (execute(poli, option));
	system("pause"); // For Visual Studio use only!
	return 0;
}
int menu() {
	int option = 0;
	cout << "Menu:" << endl;
	cout << "1. View all courses in the catalog" << endl;
	cout << "2. Add a course to the catalog" << endl;
	cout << "3. Remove a course from the catalog" << endl;
	cout << "4. Exit the program" << endl;
	cout << "Enter your choice: ";
	cin >> option;
	return option;
}
bool execute(Catalog& catalog, const int option) {
	switch (option) {
	case 1:
		// View all courses in the catalog
		if (catalog.getCount() == 0) {
			cout << "No courses have been added to the catalog yet." << endl
<<
				endl;
		}
		else {
			catalog.printCatalog();
		} break;
	case 2: {
		// Add a course to the catalog
		// Get input for code
		printf("Enter course code: ");
		cin.ignore();
		getline(cin, code);
		// Get input for name
		printf("Enter course name: ");
		getline(cin, name);
		// Get input for credits
```

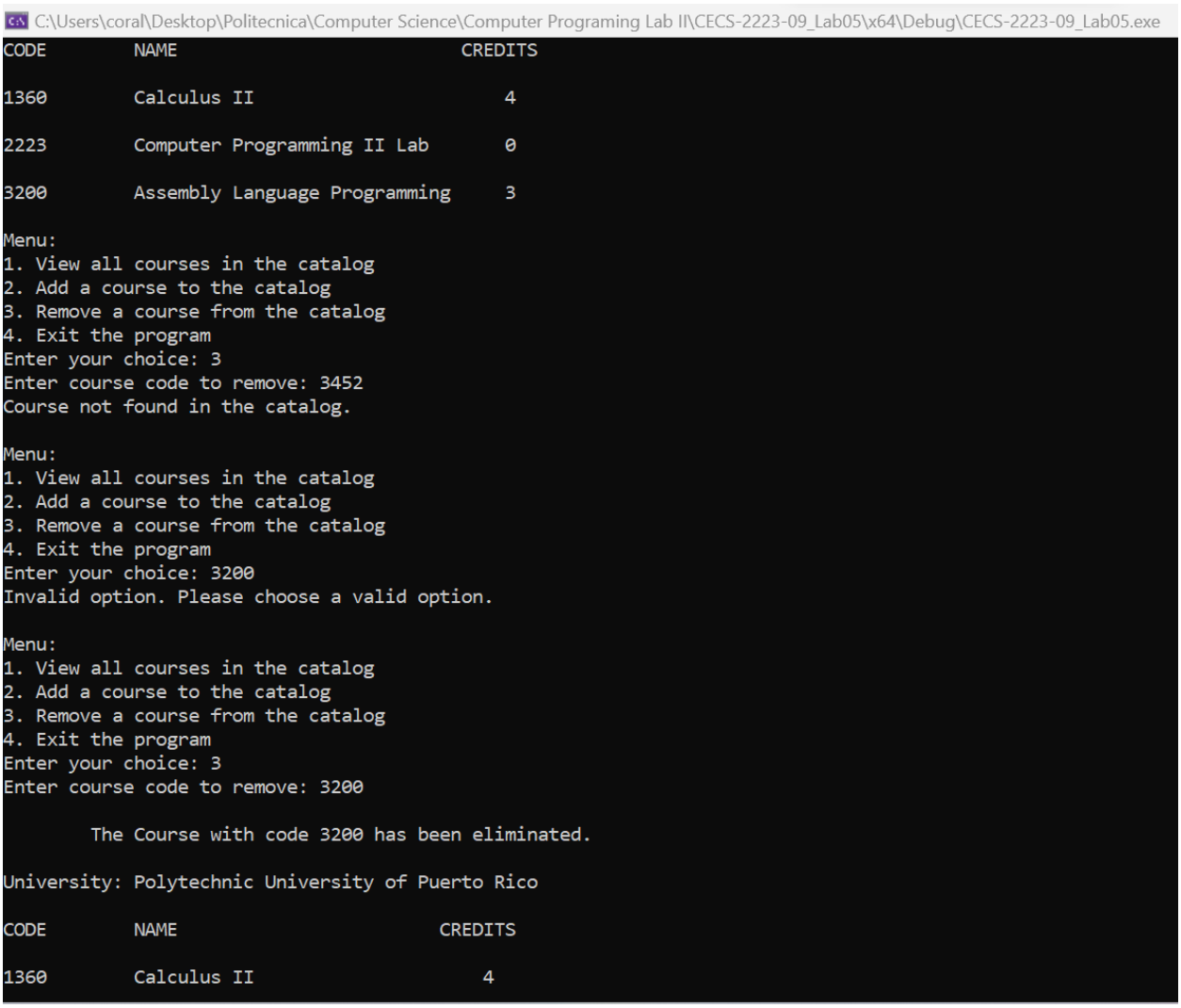
```
			printf("Enter course credits: ");
			cin >> credits;
			cout << endl << endl;
			catalog.addCourse(code, name, credits);
			catalog.sortCatalog(); // Sort the catalog after adding
		} break;
		case 3: {
			// Remove a course from the catalog
			printf("Enter course code to remove: ");
			cin.ignore();
			getline(cin, code);
			int index = catalog.findCourse(code);
			if (index != -1) {
				catalog.deleteCourse(index);
				catalog.sortCatalog(); // Sort the catalog after removal
				catalog.printCatalog(); // Print the updated catalog
			}
			else {
				printf("Course not found in the catalog.\n\n");
			}
		} break;
		case 4: {
			return false; // Exit the program
		}
		default:
			printf("Invalid option. Please choose a valid option.\n\n");
			break;
		} return true;
}
void personalInfo() {
	printf("Program developed by Aleysha M. Rivera Cores and Coral S. Schmidt
Montilla.\n\n");
}
//Edit the solution developed for Lab 5 as follows:
//Make sure that duplicate courses can't be added
//Overload the greater than operator for use in the sorting method
//Change the sorting algorithm to Selection Sort
//Use the overloaded assignment operator in the copy constructor
//Test your code as follows:
//Add at least 4 courses
//Remove at least 2 courses
//Try adding a duplicate course
//Try removing a non-existent course
//Call the print method when the array is empty
//Use the CECS-2223_Lab6t.docx document to complete your lab report.*/
```

2. Paste the screenshots of the program's execution below. (*5 points*)



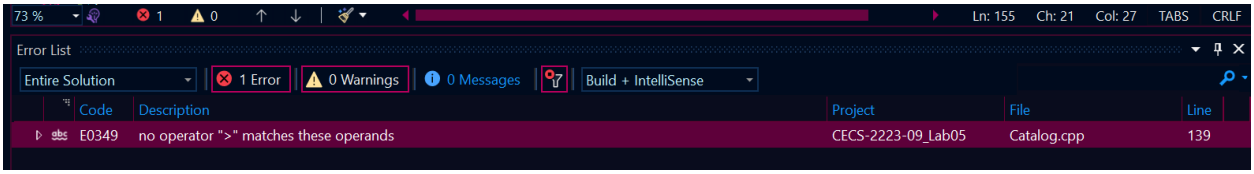

C:\Users\coral\Desktop\Politecnica\Computer Science\Computer Programing Lab II\CECS-2223-09_Lab05\x64\Debug\CECS-2223-09_Lab05.exe
Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 2
Enter course code: 2223
Enter course name: Computer Programming II Lab
Enter course credits: 0
Course with code 2223 added to the catalog.
Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 2
Enter course code: 1360
Enter course name: Calculus II
Enter course credits: 4
Course with code 1360 added to the catalog.
Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 1
University: Polytechnic University of Puerto Rico
CODE NAME CREDITS
1360 Calculus II 4
2222 Computer Programing II 4

```
C:\Users\coral\Desktop\Politecnica\Computer Science\Computer Programing Lab II\CECS-2223-09_Lab05\x64\Debug\CECS-2223-09_Lab05.exe
4. Exit the program
Enter your choice: 1
University: Polytechnic University of Puerto Rico

CODE        NAME                          CREDITS

1360        Calculus II                       4

2222        Computer Programing II            4

2223        Computer Programming II Lab       0

3200        Assembly Language Programming     3

Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 2
Enter course code: 2222
Enter course name: Computer Programing II
Enter course credits: 4


Course with code 2222 already exists in the catalog. Duplicate course not added.

Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 3
Enter course code to remove: 2222

        The Course with code 2222 has been eliminated.

University: Polytechnic University of Puerto Rico

CODE        NAME                          CREDITS
```


C:\Users\coral\Desktop\Politecnica\Computer Science\Computer Programing Lab II\CECS-2223-09_Lab05\x64\Debug\CECS-2223-09_Lab05.exe
CODE NAME CREDITS
1360 Calculus II 4
2223 Computer Programming II Lab 0
3200 Assembly Language Programming 3
Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 3
Enter course code to remove: 3452
Course not found in the catalog.
Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 3200
Invalid option. Please choose a valid option.
Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 3
Enter course code to remove: 3200
The Course with code 3200 has been eliminated.
University: Polytechnic University of Puerto Rico
CODE NAME CREDITS
1360 Calculus II 4

```
C:\Users\coral\Desktop\Politecnica\Computer Science\Computer Programing Lab II\CECS-2223-09_Lab05\x64\Debug\CECS-2223-09_Lab05.exe

Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 3
Enter course code to remove: 3200

        The Course with code 3200 has been eliminated.

University: Polytechnic University of Puerto Rico

CODE        NAME                        CREDITS

1360        Calculus II                     4

2223        Computer Programming II Lab     0

Menu:
1. View all courses in the catalog
2. Add a course to the catalog
3. Remove a course from the catalog
4. Exit the program
Enter your choice: 4
Press any key to continue . . .
```

3. Comment on any warnings or errors revealed by Visual Studio. If any error messages were present, list the error and describe how you corrected it. If no errors or warnings were revealed, comment on the most important aspect of developing the solution. (*5 points*)



We fixed it by adding :

```
bool operator>(const Course& other) const;
```

&

```
bool Course::operator>(const Course& other) const {
return getCourseName() > other.getCourseName();
}
```

To Course.cpp and Course.h