

Lab 5.1 Accessing Pointers to Classes and Structs

To assign a value to a component of object using a pointer type, use the following syntax:

```
(*ptrVariableName).classMemberName = value;
```

The dot operator (.) has a higher precedence than the dereferencing operator, requiring the use of parentheses. C++ provides another operator, called the member access operator arrow (->). To assign a value to a component of object using the -> operator, use the following syntax:

```
ptrVariableName->classMemberName = value;
```

If you want to initialize a pointer variable, the only number that can be directly assigned to a pointer variable is 0 or NULL.

Objectives

In this lab, you declare a pointer to a class, initialize the pointer to NULL, and use pointer notation to call member methods of a class.

After completing this lab, you will be able to:

- ▶ Declare a pointer to a class.
- ▶ Initialize a pointer to NULL.
- ▶ Use pointer notation to call member methods of a class.

Accessing Pointers to Classes and Structs

In the following exercises, you create a UML diagram and implementation file, and then write a program that uses pointer notation.

1a. Design a UML diagram for a class named **Grade.h** with the following private data members:

- ▶ char letterGrade;
- ▶ int numericGrade;
- ▶ string student;

The Grade class includes the following private member method:

- ▶ void calcLetterGrade(); to calculate a letter grade depending on the numeric grade, assume ≥ 90 A, ≥ 80 B, etc.

The Grade class includes the following public member methods:

- ▶ char getLetterGrade(); to return the letterGrade
- ▶ int getNumericGrade(); to return the numericGrade
- ▶ string getStudent() to return the student
- ▶ a constructor that accepts a student name and an integer numeric grade with which to initialize a new Grade object.

- 1b. Implement your design in C++ and save the class as **Grade.h**
- 1c. Write the implementation file **Grade.cpp** for the class Grade, and then compile and save the files.
- 1d. Write a C++ program to test your class that uses pointer notation and dynamic memory allocation to the class Grade, and name the program **TestGrade.cpp**. Your test driver should first ask how many students are to be entered by the user. With this number, dynamically create an array of *pointers* to Grade objects. Then in a loop, dynamically create Grade objects, asking for the name of the student and their numeric score, and using this information to dynamically create and construct a new Grade object with the appropriate information. Once created, the newly constructed Grade object should be saved in the array of grades. After entering all of the grading information, print out a summary of the students and their grades.
- 1e. Enter, compile, link and execute **TestGrade.cpp**

The following is a copy of the screen results that might appear after running your program. Input by the user is shown in bold.

This program uses pointer notation and dynamic memory allocation to the class Grade.

How many student's data will be entered: **2**

Enter Student #1's name: **Zel da Goldstei n**

Enter Student #1's numeric score: **86**

Enter Student #2's name: **Harol d Loyd**

Enter Student #2's numeric score: **93**

#	Name	Score	Grade
1)	Zel da Goldstei n	86	B
2)	Harol d Loyd	93	A

Lab 5 Finished

You have now completed Lab 5. Upload your **Grade.h**, **Grade.cpp**, and **TestGrade.cpp** in a new Lab5 folder in your online student account.

Attached at the end of this lab is a description of your fifth programming assignment, which will be due next Friday Dec 9th. After completing this lab, now is a good time to begin thinking about how you will implement the second programming assignment.