OBJECT ORIENTED PROGRAMMING 1 LABORATORY
Experiment # 4:
Classes I

OBJECTIVES

The main purpose of this experiment is to introduce you to selection and repetition statements in class design. Also function overloading concept will be analyzed.

QUESTIONS

We have examined a GradeBook class example in the course. In the GradeBook class, we have studied a constructor with a string argument, setCourseName, getCourseName, and displayMessage member functions within public access specifier. Also, a private data member in string type has been declared to store course name. We have implemented GradeBook class by separating class definition (in other words interface), class implementation and driver program which contains main function [1]. These three files are given the following figures (Figures 1-3).

1) Open a new project and implement the following programs.
2) Modify GradeBook class as follows:

a) Include *determineClassAverage* public member function which receives the number of students and returns nothing. In the function, input grades from the user, calculate and print the average of the grades. Test your class for 5 students (**Hint:** You can get help from your textbook on pages 117-118 [1]).

b) Include overloaded *determineClassAverage* public member function which receives nothing and returns nothing. In the function, input grade until the user enters a sentinel value such as -1. Calculate and print the average of the grades (**Hint:** You can get help from your textbook on pages 126-127 [1]).

c) Include *inputGrades* public member function which receives nothing and returns nothing. In the function, an arbitrary number of letter grades is read from the user by using sentinel-controlled repetition and updates the appropriate grade counter for each grade entered. You have to declare five additional private data member for each grade (A, B, C, D, and F) (**Hint:** You can get help from your textbook on pages 165-167 [1]).

d) Include *displayGradeReport* public member function which receives nothing and returns nothing. The function prints a report containing the number of students who received each letter grade (**Hint:** You can get help from your textbook on pages 165-167 [1]).

3) Draw UML Class Diagram for resulting class by using Visio.

E-mail : burakaleci@gmail.com

```cpp
/* ********************************************************
 * Filename     :   GradeBook.h
 * Author       :   Burak Kaleci
 * Date         :   27.09.2018
 * Description  :   GradeBook Class Definition
 * ********************************************************/

// program uses C++ standard string class
#include <string>
using namespace std;

// GradeBook class definition
class GradeBook
{
public:

    // constructor that initializes courseName
    GradeBook(string);
    // function that sets the course name
    void setCourseName(string);
    // function that gets the course name
    string getCourseName();
    //  function that displays a welcome message
    void displayMessage();

private:
    // course name for this GradeBook
    string courseName;
};

```

Fig. 1 GradeBook Class Definition

```cpp
/* ********************************************************
 * Filename     :   GradeBook.cpp
 * Author       :   Burak Kaleci
 * Date         :   27.09.2018
 * Description  :   GradeBook Class Implementation
 * ********************************************************/

// includes input/output stream header
// To output data to the screen
#include <iostream>
#include "GradeBook.h" // include definition of class GradeBook
using namespace std;


// constructor initializes courseName with string supplied as argument
GradeBook::GradeBook(string name)
{
    setCourseName(name); // call set function to initialize courseName
}

// function that sets the course name
void GradeBook::setCourseName(string name)
{
    if (name.length() <= 30) // if name has 30 or fewer characters
        courseName=name; // store the course name in the object
    else {
        // set courseName to first 30 characters of parameter name
        courseName =name.substr( 0, 30 ); // start at 0,length of 30
        cout << "Name \"" << name<< "\" exceeds maximum length (30).\n"
<< "Limiting courseName to first 30 characters.\n" << endl;
    }
}

// function that gets the course name
string GradeBook::getCourseName()
{
    return courseName; // return the object's courseName
}

//  function that displays a welcome message
void GradeBook::displayMessage()
{
    cout << "Welcome to the Grade Book for " << getCourseName() << "!" << endl;
}
```

Fig. 2 GradeBook Class Implementation

```cpp
/* ******************************************************
 * Filename     :   main.cpp
 * Author       :   Burak Kaleci
 * Date         :   27.09.2018
 * Description  :   Driver Program for GradeBook Class
 * ******************************************************/

// includes input/output stream header
// To output data to the screen
#include <iostream>
#include "GradeBook.h" // include definition of class GradeBoo
using namespace std;

// function main begins program execution each program must include main function
main()
{

    // create two GradeBook objects
    GradeBook gradeBook1("EEE102_ComputerProgramming_in_C");
    GradeBook gradeBook2("EEE103_AdvancedProgramming");

    // display initial value of courseName for each GradeBook
    cout << "gradeBook1 created for course: "  << gradeBook1.getCourseName()
        << "\ngradeBook2 created for course: " << gradeBook2.getCourseName()
        << endl;

}

```

Fig. 3 Driver Program for GradeBook Class

[1] Paul Deitel and Harvey Deitel, “C++ How To Program”, Eighth Edition, Prentice Hall, 2012.