|  |
| --- |
| **Computer Engineering Department - ITU** |
| **CE101L: Object Oriented Programming Lab** |

|  |  |
| --- | --- |
| **Course Instructor: Usama Bin Shakeel** | **Dated: 07/04/2022** |
| **Teaching Assistant: Aqsa Khalid** | **Semester: Spring 2022** |
| **Lab Engineer: Nadir Abbas** | **Batch: BSCE2021** |

# **Lab 5B. Using Friend Function in Classes and Objects**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Roll number** | **Report**  **(out of 100)** | **Scaled to 10** | **Total**  **(out of 10)** |
| NIMRA MAQBOOL | BSCE21012 |  |  |  |

Checked on: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## **Objective**

The objective of this lab is to observe the basic knowledge of programming classes in C++.

## **Equipment and Component**

|  |  |  |
| --- | --- | --- |
| **Component Description** | **Value** | **Quantity** |
| Computer | Available in lab | 1 |

## **Conduct of Lab**

1. Students are required to perform this experiment individually.
2. In case the lab experiment is not understood, the students are advised to seek help from the course instructor, lab engineers, assigned teaching assistants (TA) and lab attendants.

## **Theory and Background**

A friend function of a class is defined outside that class' scope, but it has the right to access all private and protected members of the class. Even though the prototypes for friend functions appear in the class definition, friends are not member functions.

A friend can be a function, function template, or member function, or a class or class template, in which case the entire class and all of its members are friends.

**Lab Task**

**Task A [Marks: 15]**

In this task, you are required to create a class **Box** with the following data members and member functions,

***Private Data Members such as:***

width(double)

***Public Member Functions such as:***

**DefaultConstructor()** – It will display “I am automatically called because I am constructor” and set width to zero.

**ParameterizedConstructor(int w)** – It will set width

***Non Member Function such as:***

**friend void printWidth(Box &b)** – It is friend function, and it will print the width

Create object of class in main function and call member and non-member function

|  |
| --- |
| **Function.cpp:**  Box::Box() {  cout << "I am automatically called because I am constructor" << endl; //making constructor and displaying statement  width = 0.0; //storing width equal to zero }  Box::Box(double w) {  cout << "enter width = "; //storing the width taken from user in the constructor  cin >> w;  width=w; //storing w's value in width }  void printWidth(Box &b) {  cout << "the width of the box is " << b.width; //printing the width value }  **Function.h:**  class Box { private:  double width; //initializing private member width public:  Box(); //declaring the default constructor  Box(double w); //declaring parametrized constructor  friend void printWidth(Box &b); //declaring a friend function };  void printWidth(Box &b);  **main.cpp:**  int main() {  double w; //initializing  Box b(w); //calling the parametric constructor  Box(); //calling default constructor  printWidth(b); //printing width |

**Task B [Marks: 25]**

In this task, you are required to create a class **A** and friend class **B** with the following data members and member functions,

***Private Data Members of class A such as:***

a(int)

***Public Member Functions of class A such as:***

**ParametrizedConstructor(int a)** – It will set value a.

***Public Non Member Class of class A such as:***

**friend class B** – Declare class B as friend

***Private Data Members of class B such as:***

b(int)

***Public Member Functions of class B such as:***

**void showA(A &a)** – It will print private member of class A.

Create objects of class A and B in main function and print private member of A by calling public member function of class B

You can assume that the dimensions are correct for the multiplication and addition.

|  |
| --- |
| **Class1.cpp:**  using namespace std; classA::classA(int a1) {  cout << "\nenter number" << endl; //taking number from user and storing that number in a that is private member  cin >> a1;  a = a1; }  void classB::showA(classA &a) {  cout << "the member of class A = " << a.a; //displaying a of a }  **function.h:**  #include <iostream>  using namespace std;  class classB; //declaring class b  class classA { private:  int a; //a private variable of int a public:  classA(int a1); //parametrized constructor declaration   friend class classB; //declaring a friend class  };  class classB { private:  int b; //declaring a private member public:  void showA(classA &a); //declaring show function };  **Main.cpp:**  int a1; //initializing a and b int a; classA A (a1); //calling the class a classB B; //making an object B.showA(A); //calling function show return 0;  **menu:**  int opt; do {  cout << "\nwhich task do you want to perform ?" << endl;  cout << "1.task 1." << endl;  cout << "2.task 2." << endl;  cout << "3.exit" << endl;  cin >> opt;  if (opt == 1) {  double w; //initializing  Box b(w); //calling the parametric constructor  Box(); //calling default constructor  printWidth(b); //printing width  }  if (opt == 2) {  int a1; //initializing a and b  int a;  classA A(a1); //calling the class a  classB B; //making an object  B.showA(A); //calling function show  }  if (opt == 3) {  cout << "YOU CHOOSE TO EXIT" << endl;  exit(6);  } }while(opt>=1 && opt<=3); |

#### **Assessment Rubric for Lab**

**Method for assessment:**

Lab reports and instructor observation during lab sessions. Outcome assessed:

a. Ability to conduct experiments, as well as to analyze and interpret data (P) b. Ability to function on multi-disciplinary teams (A)

c. Ability to use the techniques, skills, and modern engineering tools necessary for engineering practice (P)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Performance metric** | **Task** | **CLO** | **Description** | **Max marks** | **Exceeds expectation** | **Meets expectation** | **Does not meet expectation** | **Obtained marks** |
| 1. Realization of experiment (a) | 1 | 1 | Functionality | 40 | Executes without errors excellent user prompts, good use of symbols, spacing in output. Through testing has been completed (35-40) | Executes without errors, user prompts are understandable, minimum use of symbols or spacing in output. Some testing has been completed (20-34) | Does not execute due to syntax errors, runtime errors, user prompts are misleading or non-existent. No testing has been completed (0-19) |  |
| 2. Teamwork (b) | 1 | 3 | Group Performance | 5 | Actively engages and cooperates with other group member(s) in effective manner (4-5) | Cooperates with other group member(s) in a reasonable manner but conduct can be improved (2-3) | Distracts or discourages other group members from conducting the experiment (0-1) |  |
| 3. Conducting experiment (a, c) | 1 | 1 | On Spot Changes | 10 | Able to make changes (8-10) | Partially able to make changes (5-7) | Unable to make changes (0-4) |  |
| 1 | 1 | Viva | 10 | Answered all questions (8-10) | Few incorrect answers (5-7) | Unable to answer all questions (0-4) |  |
| 4. Laboratory safety and disciplinary rules (a) | 1 | 3 | Code commenting | 5 | Comments are added and does help the reader to understand the code (4-5) | Comments are added and does not help the reader to understand the code (2-3) | Comments are not added (0-1) |  |
| 5. Data collection (c) | 1 | 3 | Code Structure | 5 | Excellent use of white space, creatively organized work, excellent use of variables and constants, correct identifiers for constants, No line-wrap (4-5) | Includes name, and assignment, white space makes the program fairly easy to read. Title, organized work, good use of variables (2-3) | Poor use of white space (indentation, blank lines) making code hard to read, disorganized and messy (0-1) |  |
| 6. Data analysis (a, c) | 1 | 4 | Algorithm | 20 | Solution is efficient, easy to understand, and maintain (15-20) | A logical solution that is easy to follow but it is not the most efficient (6-14) | A difficult and inefficient solution (0-5) |  |
| 7. Computer use (c) | 1 | 2 | Documentation & GitHub Submissions | 5 | Timely (4-5) | Late (2-3) | Not done (0-1) |  |
|  | Max Marks (total): | | | 100 | Obtained Marks (total): | | |  |

Lab Engineer Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_