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

|  |  |
| --- | --- |
| **Course Instructor: Usama Bin Shakeel** | **Dated: 06/04/2022** |
| **Teaching Assistant: Aqsa Khalid** | **Semester: Spring 2022** |
| **Lab Engineer: Nadir Abbas** | **Batch: BSCE2021** |

# **Lab 5A. Use of Inline Member 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**

C++ provides an inline functions to reduce the function call overhead. Inline function is a function that is expanded in line when it is called. When the inline function is called whole code of the inline function gets inserted or substituted at the point of inline function call. This substitution is performed by the C++ compile at compile time. Inline function may increase efficiency if it is small.

**Lab Task**

**Task A [Marks: 15]**

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

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

x,y (integers)

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

**DefaultConstructor()** – It will display “I am automatically called because I am constructor”.

**void setXY(int a, int b)** – It will initialize x and y

**int getX()** – It will return value of x.

**int getY()** – It will return value of y.

Create object of class in main function and display value of x and y

|  |
| --- |
| **Function.h:**  class point { //making a class private:  int x;  int y; //declaring public:  void defaultConstructor();  void setXY(int a, int b);  int getX();  int getY(void); }; inline void point:: defaultConstructor() { //making a function of constructor  cout << "I am automatically called because I am a constructor" << endl; }  inline void point:: setXY(int a, int b) { //function to set values of x and y  int x;  int y;  cout << "please enter number 1 = ";  cin >> a;  cout << "please enter number 2 = ";  cin >> b;  x = a;  y = b;  cout << endl; }  inline int point:: getX() {  int x;  return x; //returning value }  inline int point::getY(void) {  int y;  return y; //returning value }  **main :**  do {  cout << "WHICH TASK ?" << endl;  cout << "1.TASK 1." << endl;  cout << "2.TASK 2." << endl;  cout << "3.exit" << endl;  cin >> opt;  if (opt == 1) {  point p1;  p1.defaultConstructor();  int a;  int b;  p1.setXY(a, b);  p1.getX(); //calling  p1.getY();  } |

**Task B [Marks: 25]**

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

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

a,b,add,sub,mul(int)

div(float)

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

**DefaultConstructor()** – It will display “Operation Started”.

**void get()** – It will take input a and b.

**void sum()** – It will add a and b store in add and print add.

**void difference()** – It will subtract a and b store in sub and print sub.

**void product()** – It will take product of a and b store in mul and print mul.

**void division()** – It will divide a and b store in div and print div.

**void Destructor()** – It will display “Operation Closed”.

Create object of class in main function and call all member functions

|  |
| --- |
| **Function.h**:  class operation { private:  int a;  int b;  int add;  int sub; //initializing  int mul;  float div; public:  void DefaultConstructor();  void get();  void sum();  void difference();  void product();  void division();  void destructor();  }; inline void operation:: DefaultConstructor() {  cout << "Operation Started" << endl; }  inline void operation:: get() {  int a;  int b;  cout << "please enter number 1 = ";  cin >> a; //getting a and b  cout << "please enter number 2 = ";  cin >> b;  cout << endl; }  inline void operation:: sum() {  int a;  int b;  int add; //taking sum  add = a + b;  cout << "sum of " <<a<<" and "<<b<<" = " << add << endl; }  inline void operation:: difference() {  int a;  int b;  int sub; //subtracting  sub = a - b;  cout << "sub of " <<a<<" and "<<b<<" = " << sub << endl; }  inline void operation:: product() {  int a;  int b;  int mul; //taking multiplication  mul = a \* b;  cout << "multiplication of "<<a<<" and "<<b<<" = " << mul << endl; }  inline void operation::division() {  float a;  float b; //dividing  float div;  div = a / b;  cout << "division = " << div << endl; }  inline void operation::destructor() {  cout << "Operation closed" << endl; //displaying  cout << endl; }  **main.cpp:**  if (opt == 2) {  operation op;  op.DefaultConstructor();  op.get();  op.sum();  op.difference();  op.product(); //calling  op.division();  op.destructor();  }  if (opt == 3) {  cout << "YOU CHOOSE TO EXIT" << endl;  exit(3);  }  } while (opt >= 1 && opt <= 3);  return 0; } |

#### **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: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_