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

|  |  |
| --- | --- |
| **Course Instructor: Usama Bin Shakeel** | **Dated: 16/06/2022** |
| **Teaching Assistant: Aqsa Khalid** | **Semester: Spring 2022** |
| **Lab Engineer: Nadir Abbas** | **Batch: BSCE2021** |

# **Lab 14B. Problem Based Learning in Java OOP**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Roll number** | **Report**  **(out of 100)** | **Scaled to 10** | **Total**  **(out of 10)** |
| Muhammad Abubakar Saif | BSCE21017 |  |  |  |

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**

**Java** is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. Java is a popular programming language. Java is used to develop mobile apps, web apps, desktop apps, games and much more.

**Lab Task**

**Task A: [Marks: 20]**

Write a program to create a class named shape. In this class we have three sub classes circle, triangle and square each class has two member function named draw () and erase (). Create these using polymorphism concepts

|  |
| --- |
| *//##################### TASK 1 ############################* **class** Shape {  **public void** draw() {  System.***out***.println(**"I am draw() from Base Class"**);  }   **public void** erase() {  System.***out***.println(**"I am erase() from Base Class"**);  } }  **class** Circle **extends** Shape {  **public void** draw() {  System.***out***.println(**"I am draw() from Circle Class"**);  }   **public void** erase() {  System.***out***.println(**"I am erase() from Circle Class"**);  } }  **class** Triangle **extends** Shape {  **public void** draw() {  System.***out***.println(**"I am draw() from Triangle Class"**);  }   **public void** erase() {  System.***out***.println(**"I am erase() from Triangle Class"**);  } }  **class** Square **extends** Shape {  **public void** draw() {  System.***out***.println(**"I am draw() from Square Class"**);  }   **public void** erase() {  System.***out***.println(**"I am erase() from Square Class"**);  } }  **public class** Main {  **public static void** main(String[] args) {  String rep;  Scanner obj = **new** Scanner(System.***in***);  System.***out***.println(**"Enter Credentials of first Employee: "**);  System.***out***.println(**"Enter Employee First Name: "**);  String name = obj.next();  System.***out***.println(**"Enter Employee Second Name: "**);  String name1 = obj.next();  System.***out***.println(**"Enter Employee Monthly Salary: "**);  **int** salary = obj.nextInt();  Employee worker = **new** Employee(name, name1, salary);  **double** month = worker.calculateYearlySalary(0);   System.***out***.println(**"Enter Credentials of Second Employee: "**);  System.***out***.println(**"Enter Employee First Name: "**);  String name2 = obj.next();  System.***out***.println(**"Enter Employee Second Name: "**);  String name3 = obj.next();  System.***out***.println(**"Enter Employee Monthly Salary: "**);  **int** salary1 = obj.nextInt();  Employee worker1 = **new** Employee(name2, name3, salary1);  **double** month1 = worker1.calculateYearlySalary(0);   System.***out***.print(**"The Yearly Salary of "** + worker.getFirstName() + **" "** + worker.getLastName() + **" is: "** + month);  System.***out***.println(**", while its Monthly Salary is: "** + worker.getMonthSalary());  System.***out***.println(**"The Increased Yearly Salary of "** + worker.getFirstName() + **" "** + worker.getLastName() + **" is: "** + worker.calculateYearlySalary(1));   System.***out***.print(**"\n The Yearly Salary of "** + worker1.getFirstName() + **" "** + worker1.getLastName() + **" is: "** + month1);  System.***out***.println(**", while its Monthly Salary is: "** + worker1.getMonthSalary());  System.***out***.println(**"The Increased Yearly Salary of "** + worker1.getFirstName() + **" "** + worker1.getLastName() + **" is: "** + worker1.calculateYearlySalary(1));  }  } |

**Output:**

**Text, letter

Description automatically generated**

**Text

Description automatically generated**

**Text

Description automatically generated**

**Text

Description automatically generated**

**Task B: [Marks: 20]**

Create a class called Employee that includes three pieces of information as instance variables—a first name (typeString), a last name (typeString) and a monthly salary (double). Your class should have a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the monthly salary is not positive, set it to 0.0. Write a test application named EmployeeTest that demonstrates class Employee’s capabilities. Create two Employee objects and display each object’s yearly salary. Then give each Employee a 10% raise and display each Employee’s yearly salary again.

|  |
| --- |
| *//##################### TASK 2 ############################* **class** Employee {  **private** String **firstName**, **lastName**;  **private double monthSalary**;   **public** Employee(String firstName, String lastName, **double** monthSalary) {  **this**.**firstName** = firstName;  **this**.**lastName** = lastName;  **if** (monthSalary > 0) {  **this**.**monthSalary** = monthSalary;  } **else this**.**monthSalary** = 0.0;  }   **public double** calculateYearlySalary(**int** choice) {  **double** test;  **double** year = (12 \* **monthSalary**);  **if** (choice == 1) {  test = 0.1 \* year;  year = test + year;  }  **return** year;  }   **public void** setFirstName(String firstName) {  **this**.**firstName** = firstName;  }   **public void** setLastName(String lastName) {  **this**.**lastName** = lastName;  }   **public void** setMonthSalary(**double** monthSalary) {  **if** (monthSalary > 0) {  **this**.**monthSalary** = monthSalary;  } **else this**.**monthSalary** = 0.0;  }   **public double** getMonthSalary() {  **return monthSalary**;  }   **public** String getFirstName() {  **return firstName**;  }   **public** String getLastName() {  **return lastName**;  }  }  *// ################# DRIVER FUNCTION ###########################* **public class** Main {  **public static void** main(String[] args) {  String rep;  Scanner obj = **new** Scanner(System.***in***);  System.***out***.println(**"Enter Credentials of first Employee: "**);  System.***out***.println(**"Enter Employee First Name: "**);  String name = obj.next();  System.***out***.println(**"Enter Employee Second Name: "**);  String name1 = obj.next();  System.***out***.println(**"Enter Employee Monthly Salary: "**);  **int** salary = obj.nextInt();  Employee worker = **new** Employee(name, name1, salary);  **double** month = worker.calculateYearlySalary(0);   System.***out***.println(**"Enter Credentials of Second Employee: "**);  System.***out***.println(**"Enter Employee First Name: "**);  String name2 = obj.next();  System.***out***.println(**"Enter Employee Second Name: "**);  String name3 = obj.next();  System.***out***.println(**"Enter Employee Monthly Salary: "**);  **int** salary1 = obj.nextInt();  Employee worker1 = **new** Employee(name2, name3, salary1);  **double** month1 = worker1.calculateYearlySalary(0);   System.***out***.print(**"The Yearly Salary of "** + worker.getFirstName() + **" "** + worker.getLastName() + **" is: "** + month);  System.***out***.println(**", while its Monthly Salary is: "** + worker.getMonthSalary());  System.***out***.println(**"The Increased Yearly Salary of "** + worker.getFirstName() + **" "** + worker.getLastName() + **" is: "** + worker.calculateYearlySalary(1));   System.***out***.print(**"\n The Yearly Salary of "** + worker1.getFirstName() + **" "** + worker1.getLastName() + **" is: "** + month1);  System.***out***.println(**", while its Monthly Salary is: "** + worker1.getMonthSalary());  System.***out***.println(**"The Increased Yearly Salary of "** + worker1.getFirstName() + **" "** + worker1.getLastName() + **" is: "** + worker1.calculateYearlySalary(1));  } } |

**Output:**

****

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