FEU Institute of Technology
COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

COMPUTER SCIENCE DEPARTMENT

CS0053

(PROGRAMMING TOOLS AND TECHNIQUES)

EXERCISE

5

GUI-Based Grading System Module

Name: ALBUFERA, Joshua AMBONG, Darren Jake BESAGAS, John Paul	Professor: Dr. Beau Gray Habal
Date Performed : October 9, 2024	Date Submitted: October 15, 2024

I. OBJECTIVES

At the end of this exercise, students must be able to:

Cognitive

a.) Understand the topics they have learned from lesson 5.

.

Psychomotor:

a.) Apply different programming paradigms.
b.) Create a GUI-based Java program using programming paradigm.

Affective

a.) Appreciate the concept behind this exercise.

II. BACKGROUND INFORMATION

In order to accomplish this exercise, the student must have a clear understanding of the following topics:

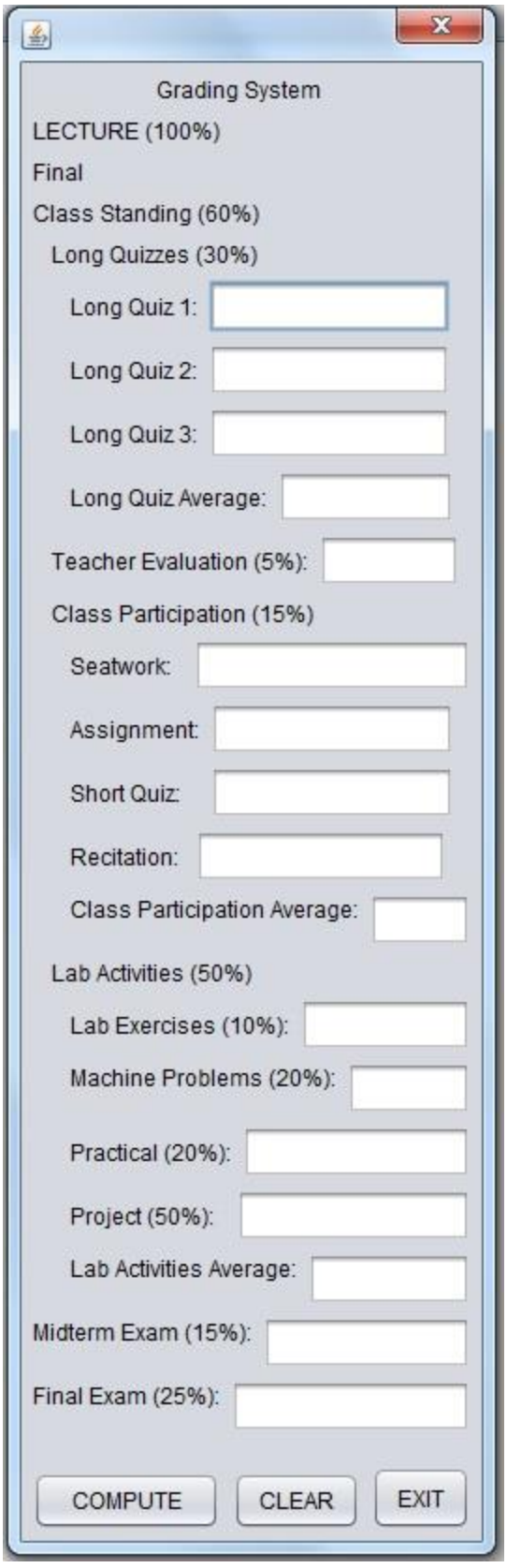
- Programming Paradigms
- try-catch statements
- Import packages or libraries

III. LABORATORY PROCEDURE

1. Create a new program.

Program Name: GradingSystem.java

2. Design your layout as shown below


Grading System
LECTURE (100%)
Final
Class Standing (60%)
Long Quizzes (30%)
Long Quiz 1:
Long Quiz 2:
Long Quiz 3:
Long Quiz Average:
Teacher Evaluation (5%):
Class Participation (15%)
Seatwork:
Assignment:
Short Quiz:
Recitation:
Class Participation Average:
Lab Activities (50%)
Lab Exercises (10%):
Machine Problems (20%):
Practical (20%):
Project (50%):
Lab Activities Average:
Midterm Exam (15%):
Final Exam (25%):
COMPUTE
CLEAR
EXIT

3. Requirements
    - The program will only accept 0-100.
    - The program will display notification info: Computed Final Grade, Equivalent Grade and Remarks: Passed/Failed/Invalid.
    - Apply coding conventions.
    - The program must be free from any errors.

IV. QUESTION AND ANSWER

1. Discuss the programming paradigms you used in the program.

   In our program, we primarily used Object-Oriented Programming in our program to organize it into distinct classes. The main class, GradingSystem_T5, handles both the logic and the graphical user interface. Each GUI component, such as buttons and text fields, is treated as an object, encapsulating its behavior and properties. This object-based approach enhances modularity, making the code more maintainable and reusable. Organizing the program around objects helped us separate concerns and ensure a clear, scalable structure throughout the application.

2. Explain the significance and the difference of each paradigm you use.

   The use of Object-Oriented Programming in our project helps to organize the code around objects, making it more reusable and easier to maintain. By encapsulating data and behavior within objects, we can efficiently manage the GUI and grading logic. OOP enables us to create modular components representing real-world elements, promoting reusability and improving maintainability. This approach ensures that the program is logically structured and scalable, with clear interactions between its components.

V. QUESTION AND ANSWER

Department	Computer Science
Subject Code	CSSSPEC2
Description	Programming Tools and Techniques
Term/Academic Year	1st Term SY 2016-2017

Topic	Programming Paradigms
Lab Activity No	5
Lab Activity	**GUI-Based Grading System Module**
CLO	**1, 2**

Note: The following rubrics/metrics will be used to grade students' output in the lab exercise 5.

Criteria	Exceptional	Acceptable	Amateur	Unsatisfactory
Specifications (40%)	The program works and meets all of the specifications. (40)	The program works and produces the correct results and displays them correctly. It also meets most of the other specifications. (35-39)	The program produces correct results but does not display them correctly. (30-34)	The program is producing incorrect results. (20-29)
Design (15 %)	The design is exceptionally attractive. Program is "user-friendly" with informative and consistent prompts and messages. (15)	The design is fairly attractive. Program is "user-friendly" with informative and consistent prompts and messages. (13-14)	The design is fairly attractive. Program is not "user-friendly" but still provide informative and consistent prompts and messages. (10-12)	The design is unattractive and not user-friendly (8-9)
Efficiency (20%)	The code is extremely efficient without sacrificing readability and understanding. (20)	The code is fairly efficient without sacrificing readability and understanding. (17-19)	The code is brute force and unnecessarily long. (14-16)	The code is huge and appears to be patched together. (10-13)
Readability (10 %)	The code is exceptionally well organized and very easy to follow. (10)	The code is fairly easy to read. (8-9)	The code is readable only by someone who knows what it is supposed to be doing. (6-7)	The code is poorly organized and very difficult to read. (4-5)
Delivery (15%)	The program was delivered on time. (15)	The program was delivered within a day of the due date. (13-14)	The code was within 2 days of the due date. (10-12)	The code was within a week of the due date. (8-9)
Total: 100%				