Premier University, Department of CSE
Spring 2024, 3rd Semester, Assignment, May 28, 2024
Course Title: Object Oriented Programming**, Course Code:** CSE 1115
Course Outcome:CO3, Total Marks: 10

Introduction: This is a take home assessment to evaluate your understanding of object oriented concepts taught in Object Oriented Programming Course. The assessment will be composed of four (4) parts: the scenario, the design, the code and explanation (including proof of testing).

The scenario:

You will decide on the topic for your project. You should choose a business for example, a café. You will write the basic use cases for the business, including identifying the major functionality the application must provide. At a minimum, your scenario needs to add objects to collections, retrieve a specific object from a collection (using list lookup) and display all objects in each collection.

The design:

You will create a simplified class diagram for your application, which shows the attributes/methods for each class and the inheritance. You should aim to have between 6 and 8 classes for your application. The classes should be: Main/driver class, parent class (at least 1), child classes (at least 2), group class (at least 1). You will explain how your design will provide the needed functionality for the application.

The code:

You will write the code for the application using the design. Make sure you follow the guidelines setout in the subject eg use camelCase for objects, LetterCase for classes, all classes have toString(). Public mutators should be used sparingly etc Your solution will need to use either interface or implementation inheritance (or both); overloading, overriding, menus with recursion and for each loops.

The explanation:

You will explain the code you wrote and any problems you encountered and how you fixed them. At minimum you need to identify: All inheritance – including why inheritance was used in this/these class(es)

All inheritance – including why inheritance was used in this/these class(es)

All overloaded methods – why overloading was used in each case

All overriding methods – why overriding was used in each case

Any polymorphism – how it affected the code

You also need to include at least 1 screen shot of your code being testing on the object bench. Please double check your code compiles and your submission document contains the updated version of all code prior to submission.

Tasks:

1. Design an object oriented application (as you choose)
2. Code proper statements for your design.
3. Explain the code with proper design rules.
4. Test at least one method.

Submission Guidelines:

- Prepare a report for documentation including Design, code, explanation and test.

- Briefly address the complex problem-solving questions:

a. Does the solution need in-depth engineering knowledge?

b. Does the solution involve wide-ranging or conflicting technical, engineering, and other issues?

c. Is the solution well-known, or does it require abstract thinking and analysis to formulate?

d. Does the solution involve infrequently encountered issues?

e. Does the solution need adherence to standards and codes of practice?

f. Does the solution involve stakeholders with conflicting technical requirements?

g. Does the solution involve interdependence between sub-problems or parts?

Rubrics for assignment marking:

Task	Criteria	Good (4-5)	Moderate (2-3)	Poor (1)
1.	Class Relationships (Problem analysis)	All major classes, relationships and inheritance have been identified. Demonstrates a clear understanding of OO concepts and how they should be applied to produce a working	Some classes and relationships have been identified and some behaviour has been placed in correct classes. Some correct inheritance. Demonstrates some	Most classes are missing or wrong. Little relationships/inheritance

		design	understanding of OO concepts and how they should be applied to produce a working design	
2.	Member Identification (Problem analysis)	Most attributes and methods have been identified correctly and placed in correct classes.	Some attributes and methods have been identified.	No attributes or methods have been identified for some classes or attributes are not valid for the system..
3.	Functional Specifications (Problem solution)	The program has been defined, works and produces correct results but may not display them correctly. It also meets most of the other specifications. Student provides clear justification for design choices	The program has been defined, is producing some correct results but a part of the program is incomplete. Student provides some justification for design choices	The program has not been well defined, is producing all incorrect results, does not run or mostly incomplete Student provides little justification for design choices
4.	Design Rules (Problem solution)	The code is easy to read, follow and is well organised. Some minor design rules or patterns have been broken. Some minor efficiency issues may be present.	The code is difficult to read at times. The flow of control is difficult to follow at times. Most design rules and patterns have not been followed.	No enough code written to assess correctness