CST8116 Lab Exam 02 Section 321

Lab Exam 2

# Instructions - Overview

* Follow the instructions starting on page 2 of this handout to design a problem solution, create a Java program, and test your program. Documenting each part using an MS Word document.
* This exam must be completed and submitted by the end of your lab period.
* Note: Each lab section will be given a unique problem statement to help detect cheating.

# Instructions – Submission Notes

* Upload your MS Word document, as well as your Java source code files (.java files) to Brightspace by the end of the lab period.
* You are not required to copy source code into your MS Word file, however your .java file(s) must be uploaded.
* Your lab professor may specify additional submission requirements in the lab period.

# Instructions – Open Book Exam

* This is an open book lab exam; you may use your textbooks, lecture notes, exercises, assignments, hybrids for help.
* You are not permitted to talk to, or chat with other students during the lab exam.
* You are not permitted to talk about, or share materials from, your lab exam, with students in other lab sections.

# Instructions – Appropriate Use of Zoom during Exam

* To get help from the professor raise your hand (virtually) using Zoom, or send a private chat to your professor. Your professor will assign you to a break out room where they can join and then help you (verbally talk and or you may share your screen).
* Please, do not un-mute your microphone and verbally ask questions in the main Zoom session, it is disruptive to other students.

# Instructions – Microsoft Word Document Format

Suggested Headings (See template provided)

Pseudocode

Screen shots of program execution

# From Lab Exam 1 (Reference and Starter Code)

* In Lab Exam 1 you would have created the following class (or similar), a starting code file to be used is provided.
* Ounces can have decimal places, price per ounce is in dollars and cents.

|  |
| --- |
| Egg |
| -ounces:double  -pricePerOunce:double |
| +Egg()  +getOunces():double  +setOunces(ounces:double):void  +etPricePerOunce:double  +setPricePerOunce(pricePerOunce:double):void  +calculateSalesCost():double |

# Lab Exam 2 Tasks

## Part 1 Pseudocode

* Write the Pseudocode for a main method that would perform the following tasks, you will need to use a repetition structure (a loop), and a selection structure (if or case) and some constants.

A client would like a menu-driven application that runs until an exit-Egg option is selected. The other two menu choices should be edit- Egg and print- Egg. If edit is selected, the fields should be updated by interacting with the user to get values and setting them into the object. If print is selected, the result of calculateSalesCost() method should be displayed. If exit is selected, a message, “program exits” is output and the program shuts down. If any other number is entered as a menu choice, the program should display a message “invalid menu input” and then continue running.

* Output the menu options directly in method main, each time the loop repeats, with your full name.
* Use the following constants inside method main for forming the menu outputs, as well as for the logic of the program

EDIT\_EGG with value 321

PRINT\_EGG with value 642

EXIT\_EGG with value 963

* **You are not required to validate inputs from the user that would crash the program**. Assume that the user will only enter numbers with the correct data-type as needed for demonstrating the running program.

## Part 2 Write the Java code for the class

* Create an Eclipse Project named **FullName\_LabExam2\_SectionNumber**
  + For example, a student with first name “Abc” and last name “Xyz” in section 123 would name the project as:

AbcXyz\_LabExam2\_Section123

* Create a class with method main. Name the class as ***FullName*LabExam2Section*Number***
  + For example, a student with first name “Abc” and last name “Xyz” in section 123 would name the class as:

AbcXyzLabExam2Section123, with resulting file name AbcXyzLabExam2Section123.java

* Using this class with a method main, follow your pseudocode to create the working program.
* Use Java coding conventions for identifiers: class, constructor, field, variable, and method names
* Comment your code for this file only, as requested and demonstrated in the course.

# Part 3 Test the Program with Screen Shot(s)

* Use the following test actions in creating screen shots of your running program; you may demonstrate more than one test in a single screen shot.
  + Option for editing the fields input, with input for each field.
  + Option for printing report input, with output displayed
  + Invalid option for exit program, output message and program should continue
  + Option for exit program.
* See the sample run of the program as reference.

# Sample of expected program execution.

Enter Option:

321 to edit egg

642 to show sales cost

963 to exit program

Program by Stanley Pieda

**321**

Enter ounces **3.2**

Enter price per ounce **0.25**

Enter Option:

321 to edit egg

642 to show sales cost

963 to exit program

Program by Stanley Pieda

**642**

**1.1199999999999999**

Enter Option:

321 to edit egg

642 to show sales cost

963 to exit program

Program by Stanley Pieda

**900**

invalid menu input

Enter Option:

321 to edit egg

642 to show sales cost

963 to exit program

Program by Stanley Pieda

**963**

program exits

# Grading (10 Points)

|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Missing / Poor (0) | Below Expectations (1) | Meets Expectations (2) |
| Pseudocode | Missing or incorrect | Pseudocode is included in MS Word document, however there are logic mistakes and-or missing the loop and-or the selection structure(s). | Pseudocode is included in MS Word document, provides a working algorithm that meets the requirements. |
| Screen shots executing program | Missing or incorrect or missing students full name in the output. | Screen do not demonstrate all of the requested menu options, or the program crashed when receiving acceptable numerical input(s). | Screen shots demonstrate the use of the menu options; the program does not crash when receiving acceptable numerical input(s). |
| .java files: comments and conventions | Missing or incorrect. | Java coding conventions for identifiers, indentation are not fully followed.  .java file with method main contains some programmer comments but missing information or incomplete. | Java coding conventions for identifiers and indentation are fully followed.  .java file with method main contains comment header with format specified for course. Each class and method has a comment above the header with a brief statement about the class or method. |
| .java files: syntax | Missing or incorrect. | Class file(s) may have small syntax mistakes that would prevent them from compiling | Class file(s) do not have syntax mistakes and should compile. |
| .java files: logic in method main. | Missing or incorrect. | Logic for the specified class within method main not correct. | Logic for the specified class within method main is correct and meets requirements. |