CST8116 Lab Exam 01 (22S) Section 323

Lab Exam

# 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.
* 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

* Hand your paper solution to professor, as well as submit your Java source code files (.java files) and screenshot of the running program to Brightspace by the end of the lab period.
* Your lab professor may specify additional submission requirements in the lab period.

# Instructions – Open Book Exam

* The first part of this exam is done on paper. Once you have written a UML diagram or pseudocode or flowchart to your lab professor’s satisfaction (60-70%) complete, you can open Eclipse and complete the code.
* 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.

# Part 1 ON PAPER DOCUMENT YOUR PLAN

# Understand the Problem and Class Design with UML or Psuedocode or FlowChart

* The client needs a program to calculate the wholesale cost of knitting a sweater based on the ounces of the completed garment. The program needs to store the weight of the sweater in ounces, and the cost of the yarn per ounce. To calculate the wholesale cost of the sweater, multiply the weight of the sweater times the cost per ounce of yarn. Design a class based on this word problem.
* Testing values are 19.5 ounces, and yarn costs 1.25 dollars per ounce.
* Using the word problem above, design a class with the needed fields, one no parameter constructor, getters and setters for each field and a worker method. Document your design using a detailed UML Class diagram or Psuedocode or a FlowChart. It is recommended to do a UML class diagram as this is probably easier.
* A program done all in main without an object class is worth a maximum of 15%
* Have the professor sign off on your design.
* Note: The calculations used in this word problem are fictional.

# Part 2 WITH PROFESSOR OK - Write the Java code for the class IN ECLIPSE

* Make sure your professor has signed off on your plan
* Open Eclipse and follow your UML class diagram or pseudocode or flowchart and code the class you designed.
* Use Java coding conventions for identifiers: class, constructor, field, variable, and method names
* Mark fields private, constructor and methods public.
* Select and use appropriate data types for the fields as well as method parameters as needed.
* Comment your code as requested and demonstrated in the course.

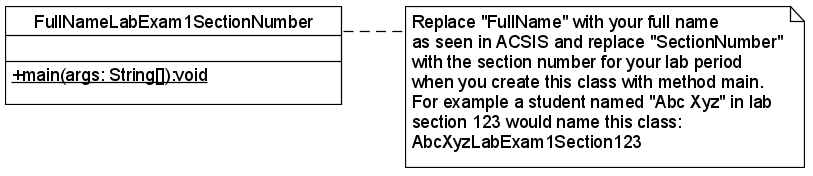
# Part 3 Test the class you created, using a main method in a separate class.

* Create a second class with an identifier that follows these instructions:
  + *FullName*LabExam1Section*Number*
  + For example, a student with first name “Abc” and last name “Xyz” in section 123 would name the class as:

AbcXyzLabExam1Section123, with resulting file name AbcXyzLabExam1Section123.java

* Using this class with a method main, write some simple program tests to verify your class.
  + Instantiate one object based on your designed class using the no parameter constructor, set a value into each field, get a value from each field printing the set value(s) on screen followed by the retrieved value(s) on screen, and show the result returned from the worker method as well.
  + There is no need for user input to the program, use literal values in the source code of your test program (within method main). Ensure that your full name is written out on screen after the tests above are run.
* Use the testing values given above as part of the word problem.

# UML Class Diagram for testing class



# Sample of expected program output (your program should be similar)

getOunces() is: 19.5

getCostPerOunce() is: 1.25

Testing worker method: 24.375

Program by Abc Xyz

# Grading (10 Points)

|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Missing / Poor (0) | Below Expectations (1.5) | Meets Expectations (3) |
| Design Plan  (on paper – UML or Psuedocode or Flowchart) | Missing or incorrect or not the specified class. | Design is partially complete and either not object oriented or missing calls to constructors/getter/setter and worker method | Design shows an object oriented solution which uses a no-arg constructor as well as getters/setters and a worker method |
| Criteria | Missing / Poor (0) | Below Expectations (0.5) | Meets Expectations (1) |
| Screen shot executing program | Missing or incorrect or does not test the specified class. | Program shows output of some of the requested tests, program may not run to completion or crashes and / or student full name does not appear as part of the output. | Demonstrates the running program, program runs to completion with expected test output as well as student’s full name. |
| .java files: comments | Missing or incorrect. | .java files contain some programmer comments but missing information or incomplete. | .java files contain comment header with format specified for course. Each class method has a comment above the header with a brief statement about the 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: conventions | Missing or incorrect. | Java coding conventions for identifiers and indentation are not fully followed. | Java coding conventions for identifiers and indentation are fully followed. |
| .java files: Constructor, Gets, Sets | Missing or incorrect. | Logic for the specified class, including accessors, mutators, no parameter constructor, may have small logic mistakes. | Logic for the specified class, including accessors, mutators, no parameter constructor, have no logic mistakes. |
| .java files: Worker Method | Missing or incorrect. | Logic for the specified class worker method may have small logic mistakes. | Logic for the specified class worker method has no logic mistakes. |
| .java files: Testing class with main | Missing or incorrect. | Class does not have specified identifier, i.e. student full name, test name, and lab section number.  and / or  Method main has logic mistakes, i.e. missing all or part of: instantiate specified class with no parameter constructor, test mutators with specified sample data, and test accessors and work method via program outputs, and write out Students full name. | Class does have specified identifier, i.e. student full name and lab section number.  and  Method main has no logic mistakes, i.e. has all of instantiate specified class with no parameter constructor, test mutators with specified sample data, and test accessors and work method via program outputs, and write out Students full name. |