CST8116 Assignment 01 (22S)

Software Development Process, design and create an Object-Oriented Java Program

# Instructions

* The Software Development Process by Cay Horstmann [1] will be used as the basis for this lab assignment.

1) Understand the problem

2) Develop and Describe an Algorithm

3) Test Algorithm with Simple Inputs

4) Translate the Algorithm into Java

5) Compile and Test Your Program

# What is the problem to solve? (Bolded highlighted text added for V2 of document)

* A wood worker uses the diameter of batches of circular-table-tops all with the same size, to calculate how many can be varnished using one can of varnish given the number of coats (applications) of varnish each table needs.
* The program should accept input for the diameter of the table-tops, the number of coats of varnish for each table-top, and **then output** how many of these table-tops can be varnished using one can of varnish.
* The diameter is to be entered in inches including fractional amounts, and an integer used for the number of coats of varnish needed. The output should indicate the number of tables that can be varnished using 1 can of varnish.
* The work worker only uses one type of varnish, and 1 can of varnish (1 litre) will coat 46800 square inches of surface.
* For the purposes of this assignment, each time the program is run, the assumption is that all of the tables to be varnished will be the same size.
* For example, given a batch of table-tops with a 42.5-inch diameter, needing 4 coats of varnish each, 1 can of varnish will complete approximately 8.2474 tables.

## 1) Understand the problem (MS-Word Document Submission)

* Given the sample data above, can you determine how the calculations were made?
* Briefly outline in general steps how you would solve the problem, and include examples of math calculations.
  + You need to document how the calculations are performed using math equations, (or rather how will a computer program be able to perform the calculations which is the next parts for the assignment).
* Include this in your MS Word document.
* Tip: See the Appendix(es) at the end of this document for help.

## 2) Develop and Describe an Algorithm including Classes (MS-Word Document Submission)

## 2a) UML Class Diagrams

* Determine what objects, the properties of the objects, and behaviors would be required for an object-oriented program, start with simple UML class diagrams and as you work out more details document the design with a detailed UML class diagram.
* Include your detailed UML Class diagrams in your MS Word document. (See lecture notes week 3).
* Tip: Consider what candidate object could store data like diameter produce a surface area to be varnished, what candidate object would use a surface area and number of coats to determine how many tables can be varnished. Also include a class with a method main to run the program. The expectation for this assignment would be two to three classes.

## 2b) Pseudocode and Flowchart (MS-Word Document Submission)

* NOTE: You are expected to use the Java API Math class to obtain a value for PI, and to calculate the power of the radius (or diameter) raised to 2 (regarding radius or diameter see the appendix at the end of this document).
* Write pseudocode and flowchart for method main, as well as any method(s) that calculate surface area for a table-top, as well as the calculation for the number of tables that can be painted.
* Include your pseudocode and flowchart in your MS Word document. (See lecture notes weeks 1, 2 and 3).
* Note: You are not required to write pseucode or flowcharts for get or set methods or constructors for this assignment, the expectation would be method main and one or two worker methods.

## 3) Test Algorithm with Simple Inputs (MS-Word Document Submission)

* As per the lecture notes, use a table within your MS Word document to test the algorithm. Consider picking numbers that might be expected as input and work through the algorithm documenting expected outputs, you may use a calculator.
* If there is a problem with the algorithm based on this desk-check then correct the pseudocode and flowchart, UML class diagram(s) and repeat this step again.

## 4) Translate the Algorithm into Java

* You are to use the Eclipse IDE to create your Java program, use a project name like Assignment 01.
* See the Java API documentation for class Math as well as the appendix at the end of this document.
* Don’t forget to comment your code files, with the expected code headers
  + File-level comments, similar to an assignment cover page.
  + Class-header comment, just above the class declaration with brief description of the class
  + Method-header comment(s), just above each method or constructor with brief description
* You are not required to copy and paste code into the MS Word document, however your .java file(s) must be submitted as Java source code files along side your MS Word document.

## 5) Test Your Program (MS-Word Document Submission)

* Re-create your testing table from step 3 in this section, but document what the program outputs are using valid inputs, do they match expectations?
* Test with some invalid inputs, and document what happens. Note that some input will crash your program, this is okay as you may not know how to fix this at this point in the course. Document what the error messages are in your test plan. Suggested invalid tests: enter a String instead of a number, use a negative number as input, use zero as part of the input. Try some things and document what happens.
* **The test plan for the program must be a separate table in your document, if you provide only one test plan table for both testing the algorithm and testing the program you will not be awarded marks for both. Minimally, this second table must also have each row updated to indicate that it was used to test the program, in other words if you copy and paste an exact copy of your first table unchanged you will not earn full marks.**

## 

## Demonstration in lab period

* Demo: Your lab professor will ask you basic questions related to your program code in the lab period as well as request you compile and run the program. If you do not demonstrate this lab to a lab professor and / or you cannot answer basic questions you may lose up to all marks for this portion of the assignment at the lab professor’s decision.
* See the course CSI for details on which week the Assignment demonstration will take place, reach out to your lab professor before then with questions or concerns.

# Microsoft Word Document Format

See the template example, suggested headings below:

1) Understand the problem

Brief paragraph and sample math calculations

2) Develop and Describe an Algorithm

UML class diagrams, Pseudocode, Flowchart

3) Test Algorithm with Simple Inputs

Test plan as table

4) Test Your Program

Second test plan as table, updated and including notes on what happens with invalid inputs.

5) References:

* Document any sources you used that are either from your textbooks, or external from the lecture handouts.
* Note: You are not required to reference class Math from the Java API, it is assumed you will refer to the API documentation as part of completing this lab work.

While you are not required to copy and paste your source code inside the MS Word document, you must upload your .java file(s) in addition to the MS Word document.

# Grading (22 points)

|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Missing / Poor (0) | Below Expectations (1) | Meets Expectations (2) |
| Understand the problem | Missing or poorly done or missing demo. | Problem solution statement is partly correct, may not have examples of the calculations required to solve the problem. | Problem solution statement is correct. Sample calculations needed to solve the problem are provided. |
| Algorithm: UML class diagram | Missing or poorly done or missing demo. | Class diagram(s) are not in correct format, properties and methods may not be assigned correctly to the classes and / or there is only one class that contains all of the program functionality. | Class diagram(s) are correct format, properties and methods are assigned to appropriate classes, based on the word problem. |
| Algorithm: pseudocode | Missing or poorly done or missing demo. | Not in correct format and / or steps are not in an order that produces correct results. Not all worker methods and / or method main are documented. | Correct format, steps are in order that produces correct results. Worker methods and method main are documented. |
| Algorithm: flowchart | Missing or poorly done or missing demo. | Not in correct format and / or steps are not in an order that produces correct results. Not all worker methods and / or method main are documented. Flowchart logic may differ from pseudocode. | Correct format, steps are in order that produces correct results. Worker methods and method main are documented. Flowchart logic closely matches pseudocode. |
| Test Plan: Algorithm | Missing or poorly done or missing demo. | Does not have correct table format as seen in lecture notes and lab exercises, and / or does not test program using samples of acceptable inputs. | Has correct table format as seen in lecture and lab exercises, tests program using samples of acceptable inputs. |
| Test Plan: Program | Missing or poorly done or unchanged copy of algorithm test table, or missing demo. | Does not have correct table format as seen in lecture notes and lab exercises, and / or does not test program using samples of acceptable inputs and / or does not document program errors resulting from invalid inputs. | Has correct table format as seen in lecture and lab exercises, tests program using samples of acceptable inputs, and documents program errors resulting from invalid inputs. |
| Source Code: \*.java file(s) Comments and Conventions | Missing or poorly done or missing demo or is starter code with no modifications. | File comment header with student name is present. Class and / or class-member (constructors, methods) are missing comment headers. Loosely follows Java coding conventions for identifiers, indentation. | File comment header with student full name is present. Class and / or class-member (constructors, methods) have comment headers. Closely follows Java coding conventions for identifiers, indentation. |
| Source Code:  \*.java file(s) program structure and logic. | Missing or poorly done or missing demo. | Program may have small syntax mistakes or produces incorrect output. Program may consist entirely within method main (is not object-oriented). | Program has correct syntax and program logic that produces correct output. Program is object-oriented with classes, fields, get / set methods, worker methods. |
| Source Code: Demo | Missing or poorly done. Student cannot correctly answer basic questions on their program code. | Student answers to basic questions on their program code are partly correct. | Student can correctly answer basic questions on their program code. |
| Running Program: Demo | Missing or poorly done. Student cannot demonstrate compilation and execution of their program in Eclipse. | Student can demonstrate compilation and execution of their program in Eclipse. Program may not work correctly with input values specified by lab professor. | Student can demonstrate compilation and execution of their program in Eclipse. Program does work correctly with input values specified by lab professor. |
| Submission | Missing or missing demo. | Student does not provide both MS Word and .java file(s) with their submission, and/or does not follow lab professor’s submission requirements. | Student does provide both the MS Word document and .java file(s) with their submission, and does follow lab professor’s submission requirements. |

# Submission Requirements

* Follow your lab professor’s requirements for submission of lab work, you will need to submit your MS Word document and your Java source code file(s).

# References

[1] Cay Horstmann. (2019). Big Java Early Objects. 7th Ed. Wiley.

[2] Joyce Farrell. (2018). Programming Logic & Design Comprehensive. 9th Ed. Cengage Learning.

[3] Cue Learn Pvt. Ltd. (2021). Area of a Circle. Last accessed Dec 28, 2021. Retrieved from https://www.cuemath.com/geometry/area-of-a-circle/

# Appendix Additional Resources

* Area of a Circle? Ans from [3]: area = PI \* r2 **or** area = PI \* d2 / 4
* Where PI can be obtained using Math.PI in Java
* d is diameter
* r is radius, the radius is one-half the diameter
* \* is multiplication in Java, / is division in Java
* r2 or d2 means the value is squared or multiplied against itself once, explore Math.pow in Java.
* How many millilitres are in a litre? Ans: 1000 millilitres per 1 litre
* How many inches are in a foot? Ans: 12

# Appendix: Sample of Program Running

Enter diameter (inches): **42.5**

Enter coats needed: **4**

Tables per one can: 8.24742016520493

Program by Stanley Pieda

Note: The user inputs above were formatted with a black font color in a bolded font with yellow high-lighting, using Microsoft Word. The default color scheme in the Eclipse console is a light green for user inputs.

(You are not expected to change the output formatting of programs in this course, the formatting was changed here in this document to make this document more accessible).