CST8116 Assignment 2 (22F)

Software Development Process, Object Oriented program with decisions and loops.

# Instructions

The Software Development Process as presented 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

# Overview

* A factory produces high-quality circular saw blades.
* For quality control the company samples diameter and kerf of the produced saw blades to ensure they are within 1/64th of an inch (0.015625 inches). A kerf is the width of the cut made by the saw blade [2].
* The company wants software that will allow a worker to enter the measured diameter and kerf in inches.
* The program should determine if the diameter and kerf is each within tolerance or not and report this to the user.
* Additionally, the program should keep and report on the number of saw blades entered, and how many measurements of diameter and kerf were out of tolerance.
* Users can make mistakes, so the program should not crash if text is entered instead of numbers when expected, and the program should not permit negative measurements to be entered.
* A junior programmer started the project but could not complete it, a senior programmer reviewed the project and then provided some //ToDo comments and has now passed the project to you to complete it.

# Task 1: UML Class Diagrams

* Examine the starter code, read the ToDo comments provided.
* Document the structure of the provided classes using detailed UML class diagrams.
  + Remember, static class members should be underlined within a UML Class Diagram
  + Don’t forget to document EPSILON when you add it.
  + You are not permitted to change the structure of the provided classes or methods, if in doubt contact your lab professor.
* Place the detailed UML Class Diagrams into your MS Word document as an image, ensure your full name as in ACSIS is within the image.

# Task 2: Pseudocode and Flowcharts

Focus on four methods, which you need to complete to make the program work:

public static isDiameterInTolerance() of class CircularSawBladeVerifier

public static isKerfInTolerance() of class CircularSawBladeVerifier

public static inputPositiveDouble(String prompt) of class User

public static void main(String[] args) of class Assignment02

* Create pseudocode and flowcharts for the four methods listed above.
  + Just the four methods, you are not required to pseucode or flowchart all of the classes and other methods.
  + The existing programmer comments in the starter code files can be used to help create your pseudocode and flowcharts.
  + Your main method must use an instance of class CircularSawBlade, and use methods of class User and CircularSawBladeVerifier.
  + You may use any decision structure (if, or switch) and either a while or do-while loop in solving this assignment (A for loop is not recommended for this program)
  + **The Conditional Operator is not accepted in place of a decision structure for marks in this assignment.**
* Place the Pseudocode as text, and flowcharts as image(s), into your MS Word document.
* Ensure your name is within the flowchart image(s)
* See lecture materials weeks 9 and 10, as well as the required readings for guidance.

# Task 3: Test Algorithm with Simple Inputs

* Feel free to use the sample below as a starting point but add more tests.
* Each row should represent a separate run of the program, minimally one iteration of the main program loop.
* Suggested Tests
  + is the check to keep looping case-insensitive? Does the program end with “no”, or other e.g. “carrot”?
  + separate boundary tests for diameter and kerf
    - below tolerance, at lower tolerance, within tolerance, at upper tolerance, above tolerance
  + is the count of blades entered correct, is count of diameters and count of kerfs out of tolerance correct?
  + Does the program prevent text input for numbers, does it prevent negative input for numbers?

|  |  |  |  |
| --- | --- | --- | --- |
| Input | Expected Output | Actual Output | Description |
| 7.23  0.125 | Enter measured diameter:  Enter measured kerf:  diameter: 7.2300, width: 0.1250  Diameter out of tolerance.  Kerf out of tolerance.  Blades Entered: 1  Diameter(s) out of tolerance: 1  Kerf(s) out of tolerance: 1  Program by *Student Name*  continue(yes/no)?  enter measured diameter: | Enter measured diameter:  Enter measured kerf:  diameter: 7.2300, width: 0.1250  Diameter out of tolerance.  Kerf out of tolerance.  Blades Entered: 1  Diameter(s) out of tolerance: 1  Kerf(s) out of tolerance: 1  Program by *Student Name*  continue(yes/no)?  enter measured diameter: | A hand trace of the pseucodecode for method main, and supporting methods, shows that the logic of the algorithm has the correct sequence of steps, and that the processing continues as the user entered ‘yes’ |
|  |  |  |  |

(Around 16 to 20 tests i.e., rows in the table in total)

# Task 4: Translate the Algorithm into Java

* Follow the ToDo sections in the starter code
* Use your pseudocode and flowcharts to help work things out for the methods you have been asked to complete
* Type brief comments in the source code files for
  + Top of the file, and each class header, and each constructor and method header where indicated.
* Reminder, your program must be object oriented, must use the starter code, and your source code must be submitted.
* You may remove the instructional comments as you implement your source code, replace with your own comment sections where directed.

# Task 5: Compile and Test Your Program

* Use your Algorithm test plans and document testing your Java program, **however** **update the description column to document the program tests.** (There is no requirement to re-write all of the tests completely).
* Take a screen shot of the running program, ensuring your name appears in the screen shot as part of the program output.

# Microsoft Word Document Format

See the template example, suggested headings below:

1) UML class diagrams

2) Pseudocode and Flowcharts

3) Test Algorithm

4) Test Program

5) Screen shot of program execution

References (as needed)

# Starter Code

* The .java files are provided:

Assignment02.java

CircularSawBlade.java

CircularSawBladeVerifier.java

User.java

# Appendix: Sample of Program Run (your program output should be similar)

Enter measured diameter: **7.25**

Enter measured kerf: **0.059**

diameter: 7.2500, width: 0.0590

Diameter in tolerance.

Kerf in tolerance.

Blades Entered: 1

Diameter(s) out of tolerance: 0

Kerf(s) out of tolerance: 0

Program by Stanley Pieda

Continue (yes/no)? **yes**

Enter measured diameter: **42**

Enter measured kerf: **42**

diameter: 42.0000, width: 42.0000

Diameter out of tolerance.

Kerf out of tolerance.

Blades Entered: 2

Diameter(s) out of tolerance: 1

Kerf(s) out of tolerance: 1

Program by Stanley Pieda

Continue (yes/no)? **yes**

Enter measured diameter: **-5**

Enter positive number only.

Enter measured diameter: 7.25

Enter measured kerf: **tuna**

Invalid input. Enter a number: **.5**

diameter: 7.2500, width: 0.5000

Diameter in tolerance.

Kerf out of tolerance.

Blades Entered: 3

Diameter(s) out of tolerance: 1

Kerf(s) out of tolerance: 2

Program by Stanley Pieda

Continue (yes/no)? **no**

Program has shut down.

Notes: User input was formatted with bold font, and highlighted. However, the default settings in Eclipse use a light-green color for user input.

# Grading (16 points)

* **If you do upload all of your Java Source code files for this assignment, the entire assignment becomes zero.**

|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Missing / Poor (0) | Below Expectations (1) | Meets Expectations (2) |
| Algorithm: UML class diagrams | Missing or poorly done or missing code files. | Class diagram(s) are not in correct format, properties and methods may not be assigned correctly to the classes and / or the diagrams do not follow the starter code provided. | Class diagram(s) are correct format, properties and methods are assigned to appropriate classes, based on the word problem and provided starter code. |
| Algorithm: pseudocode | Missing or poorly done or missing code files. | Not in correct format and / or steps are not in an order that produces correct results. Use of repetition structure(s) and / or selection structure(s) partly correct. | Correct format, steps are in order that produces correct results. Use of repetition structure(s) and / or selection structure(s) is correct. |
| Algorithm: flowchart | Missing or poorly done or missing code files. | Not in correct format and / or steps are not in an order that produces correct results. Use of repetition structure(s) and / or selection structure(s) partly correct. Flowchart logic may differ from pseudocode. | Correct format, steps are in order that produces correct results. Use of repetition structure(s) and / or selection structure(s) is correct. Flowchart logic closely matches pseudocode. |
| Test Plan: Algorithm | Missing or poorly done or missing code files. | Does not have correct table format as seen in lecture notes and lab exercises, and / or does not test program using suggested tests. | Has correct table format as seen in lecture and lab exercises, tests program using suggested tests. |
| Test Plan: Program | Missing or poorly done or unchanged copy of algorithm test table, or missing code files. | Follows from algorithm test table but does not have updated descriptions for all of the tests. | Follows from algorithm test table, does have updated descriptions for all of the tests. |
| Source Code: \*.java file(s) Comments and Conventions | Missing or poorly done or missing code files 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 code files or is starter code with no modifications. | Program may have small syntax mistakes or produces incorrect output. Use of if or switch, and / or while or do-while not fully correct (has syntax, runtime, and or logic errors) | Program has correct syntax and program logic that produces correct output. Use of if or switch, and / or while or do-while is correct (no syntax, runtime, or logic errors) |
| Running Program Screen Shot | Missing or poorly done or missing student name in screen shot or missing code files. | Screen shot may not show the program running to completion. Student name is present but may not be part of program output. | Screen shot shows program running to completion, and shows student name as part of program output. |

# Submission Requirements

* Upload your MS Word document as well as your Java file(s) to the Brightspace submission area by the due date. (See Brightspace for due date).
* Follow your lab professor’s instructions regarding lab submissions for their lab section.
* Remember: you must submit your source code files, or the entire mark for this assignment becomes zero.

# References

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

# [2] C. Baylor. What Does the Term "Kerf" Mean in Woodworking? thesprucecrafts.com.

# https://www.thesprucecrafts.com/definition-of-kerf-3536563 (Accessed Oct 28, 2022).