SE/CprE 4160: Assignment #3

Revision 1

Due on May 2, 2025 at 11:59 PM

70 Points (7.5% Overall)

Late Policy

Assignments must be submitted electronically on the specified due date **Friday 5/2/2025 at 11:59 PM**. If an assignment is submitted up to two days late but before **Sunday 5/4/2025 at 11:59 PM**, a 5% penalty will apply. For submissions beyond two days late but before **Friday 5/9/2025 at 11:59 PM**, a 15% penalty will apply. *No assignments will be accepted after **Friday 5/9/2025 at 11:59 PM**.*

Submission Instructions

- **Code Submissions:** For questions requiring code, you may include shorter or key parts of your solution directly in the document (e.g., using inline code snippets). For longer code listings or complete projects, please push your code to an open-source repository (*e.g.*, GitHub, GitLab) and provide a link in your submission. This helps keep your write-up concise.
- Your program should follow the high-quality code concepts and practices we have discussed in class including (but not limited to): code readability and documentation, low coupling and high cohesion, naming convention, code style and organization, and error handling.
- When documenting your code, please provide meaningful and insightful comments to explain what operation is performed by each statement and the intuition behind it. Of course, you only need to comment on code snippets that are either ambiguous or require extra reasoning from the reader.

Problem 1

(25 points)
Develop a program in the programming language of your choice to generate and print all permutations of a given input string using an **iterative approach only**. Your implementation must not use recursion and must not rely on any external or special string-processing libraries. Only basic string operations such as concatenation, slicing, or indexing are allowed.

1. (*10 points*) Implement the program following the principles of high-quality code covered in class. This includes, but is not limited to:
   - Clear and consistent naming conventions
   - Modular structure with low coupling and high cohesion
   - Proper code formatting and organization
   - Meaningful comments that explain non-obvious logic or design decisions
   - Basic input validation and error handling where appropriate

   Your code should be well-documented and easy to read. Avoid redundant or trivial comments—focus on explaining complex or non-trivial steps in the algorithm.
2. (*5 points*) Provide a brief written explanation describing how your code reflects the above high-quality coding principles. Address specific decisions you made to improve readability, structure, and robustness.
3. (*5 points*) Now, implement the same program using **recursive approach**. Make sure to follow proper guidelines we studied in class to produce high-quality code as in the previous code.
4. (*5 points*) Provide a brief written explanation describing which implementation *iterative* or *recursive* do you prefer and discuss your opinion in terms of readability, structure, and robustness.

Problem 2

(15 points)

In this problem, you will revisit your solution to **Assignment 1, Problem 4**—where you designed and implemented a toy-sized object-oriented system demonstrating inheritance, interfaces, polymorphism, and coupling. Your task now is to **enhance the quality of your previously written code** by applying the software quality and optimization techniques discussed in class.

1. *(10 points)* Improve your original code based on the techniques we studies so far. Submit the enhanced version of your code in your repository (you may update the existing one or create a new one). Ensure that your new implementation still satisfies all the original requirements from Problem 4 from Assignment 1.
2. *(5 points)* In your answer document, provide a clearly organized list of all the enhancements you applied. For each enhancement, briefly explain:
   - What change you made
   - Why you made the change
   - The impact of the change in terms of performance, readability, cohesion, or coupling

Problem 3

(10 points)
The `SensorDataProcessor` class[1] is an example of inefficient and poorly written code. Study the code and answer the following questions:

1. (*6 points*) Suggest at least 3 strategies to speed-up the code between the lines $36 - 52$ in function `calculate`. Use line numbers to communicate your strategies.
2. (*4 points*) Perform code jamming for the code between the lines $36 - 58$ in function `calculate`.

[1] `https://gist.github.com/atamrawi/a0f0f79b76d04618ff0420eb19b97dac`

Problem 4

(25 points)

The Apache POI project[2] provides pure Java libraries for reading and writing files in Microsoft Office formats, such as Word, PowerPoint and Excel. The source code of the project can be found on Github[3]. Specifically, we are interested in the class `XSSFWorkbook`[4]. This class presents a case of poorly written code with severe problems in coupling, cohesion, style, performance, and debugging. It also lacks proper documentation and user comments. Familiarize yourself with the source code for the class `XSSFWorkbook` at the provided link and answer the questions below:

1. (*16 points*) Write a high-quality version of function `cloneSheet`[5]. Your enhanced version **must** split the function into smaller highly-cohesive functions and should follow the high-quality code concepts and practices we have discussed so far in terms of: code readability, documentation, low coupling, high cohesion, naming convention, code style and organization, and error handling. To get the full credit, you also need to specify your re-factoring strategies that you adopted to achieve the high-quality version.

2. (*9 points*) Assume that the authors for class `XSSFWorkbook` has consulted you to perform a code review. For each of the following cases, briefly discuss your recommendation and the reasoning behind it:

   (a) (*3 points*) Improving the quality of the code between the lines 368−385 in function `onDocumentRead`[6]?

   (b) (*3 points*) The sufficiency of the exceptions and error handling in function `setSheetName`[7].

   (c) (*3 points*) The quality (acceptability) of the corrective operation on line 1615[8] in function `setSheetName`.

[2] https://poi.apache.org/

[3] https://github.com/apache/poi

[4] https://github.com/apache/poi/blob/REL_4_1_1/src/ooxml/java/org/apache/poi/xssf/usermodel/XSSFWorkbook.java

[5] https://github.com/apache/poi/blob/4d9f665b7054a32c34080a6184431d32d1eb9819/src/ooxml/java/org/apache/poi/xssf/usermodel/XSSFWorkbook.java#L602

[6] https://github.com/apache/poi/blob/4d9f665b7054a32c34080a6184431d32d1eb9819/src/ooxml/java/org/apache/poi/xssf/usermodel/XSSFWorkbook.java#L360

[7] https://github.com/apache/poi/blob/4d9f665b7054a32c34080a6184431d32d1eb9819/src/ooxml/java/org/apache/poi/xssf/usermodel/XSSFWorkbook.java#L1605

[8] https://github.com/apache/poi/blob/4d9f665b7054a32c34080a6184431d32d1eb9819/src/ooxml/java/org/apache/poi/xssf/usermodel/XSSFWorkbook.java#L1615