CPSC 501 – Advanced Programming Techniques
Final Examination - April 2025

Name:____ ID:____________________________

Time limit: 120 minutes. You may use your single 8.5 by 11-inch double-sided cheat sheet (printed or written); however, no electronic devices are allowed.

Put your full name and ID on this booklet.

Answer all questions in this booklet in the space provided.

Mark distribution:

Question Type	**Number**	**Weight**	**Score**
Performance Optimization:	1	15	_______
Reflection:	2	15	_______
Reflection:	3	15	_______
Total		**45**	_______

Part 1: Performance Optimization

Edit the included **Java 23** code to do this part.

The function definition (parameters and return type)
  start(…)
should not be changed.

Please perform step-by-step code-tuning techniques to transform
  start -> part1 -> part2 -> part3 -> part4 -> part5
Each new version must be based on the previous version.
Use the **C-style code tuning** techniques discussed in **lectures** to improve the code.

Your code should be such that every call to **start(…)** in the previous code could be **swapped** with your **partX(…)** optimized version and the **result** of the code would be **unchanged** from what was previously the result of **start(…)**.

DO NOT EDIT THE NAME OF THE FUNCTION
DO NOT CREATE MORE FUNCTIONS
DO NOT IMPORT ANYTHING
JUST EDIT THE BODY OF THE FUNCTION

Each of your code tunings will be given 3 marks of the total 15. Marks are given for proper name of the code tuning that is done. **This name must be from the lecture material (slides).**
You can reference line numbers as being unchanged from a prior step, if you want, instead of re-writing identical code. You will likely have to re-number your code at each part to continue to do this later after start -> part1.

```
public static void start(...) {
      //Optimize code below
1     ...
...
12   ....
       //Optimize code above
 }
```

Version 1

Name of technique used: ______________________________

```
public static void part1(...) {
  //Optimized code below
```

Version 2

Name of technique used: ______________________________

```
public static void part2(...) {
  //Optimized code below
```

Version 3

Name of technique used: ______________________________

```
public static void part3(...) {
  //Optimized code below
```

Version 4

Name of technique used: ______________________________

```
public static void part4(...) {
  //Optimized code below
```

Version 5

Name of technique used: ______________________________

```
public static void part5(...) {
  //Optimized code below
```

Part 2: Reflection

Use **Java 23** and the **java.lang.reflect.* API** to create a class called Test, with a main() method that can handle command-line arguments. You will ignore all Exceptions from main() [i.e. do not bother with try/except concerns]. The modifiers for constructors/methods/fields will be unknown and this knowledge should be dealt with appropriately. Assume the input is correct and ignore error checking. The main() method should do the following things:

1. …
2. …
3. …
4. …
5. …
6. …
7. …
8. …
9. …

```
public class Test {
   public static void main(String[] args) throws Exception{
...
```

Part 3: Reflection

Use **Java 23** and the **java.lang.reflect.* API** to serialize an arbitrary object into (***XML or JSON will be picked***) format String stream (do **NOT** use any ***JSON/XML*** library or other library to do this). You are expected to need to make use of the packages **java.util, java.io, java.lang.reflect**

More instructions

You will serialize to a **java.io.PrintStream** object that is passed in as a parameter. Use the human-readable (text) format specified below, using only the **print()** and **println()** methods to write to the text stream. **These methods are identical to the System.out.println() functions commonly used in Java, for example stream.println("{").**

XML or JSON FORMAT re-described here

EXAMPLE

Your solution must be generic and not design for one particular object. Your solution must be completed using general reflection.

```
public static void serialize(Object obj, PrintStream s) throws Exception {
```