Tutorial/Lab 1 – OOP Review 1

Aim

This tutorial/lab aims to review basic object-oriented programming concepts, such as arrays, classes, objects, and arrays of objects.

Tips

Create a good test set first, which can cover all kinds of cases of input values. This will be helpful to test your code later after you write your solution, but also in understanding the problem statement before you start solving the problem.

Exercise 1.1 In-place Reverse

- The reverse method in the lecture reverses an array of integers by copying it to a new array and returning it.
- Write a method **reverseInPlace** that reverses the array of doubles passed in the argument and returns nothing.
- After that, write a test program that prompts the user to enter arbitrary numbers, invokes the method to reverse the numbers, and displays the array.
- Your test program should first prompt the user to enter the array size.
- Test the method on all kinds of input array.

Exercise 1.2 Deciding four consecutive numbers

- Write the following method that tests whether the array has four consecutive numbers with the same value:
  public static boolean isConsecutiveFour(int[] values)
- After that, write a test program that prompts the user to enter a series of integers and displays it if the series contains four consecutive numbers with the same value.
- Your program should first prompt the user to enter the input size—i.e., the number of values in the series.
- Test the method on all kinds of input values.

Exercise 1.3 Stopwatch to measure running time

- Design a class named **StopWatch**.
  The class contains:
  - Private data fields startTime and endTime with getter methods.
  - An empty constructor that initializes startTime and endTime with the current time.
  - A method named start() that resets the startTime to the current time.
  - A method named stop() that sets the endTime to the current time.
  - A method named getElapsedTime() that returns the elapsed time (the difference between stop and start time) for the stopwatch in milliseconds.
- Note that you can use System.currentTimeMillis() to obtain the current time.
- Write a test program that measures the execution time of sorting 100,000 numbers using ***Selection Sort*** discussed during the lecture.

Exercise 1.4 Student Records

- Create a class **Student** to represent students in an Advanced OOP course.
- Each student object should represent a first name, last name, email address, and group number.
- Include a **toString()** method that returns a string representation of a student and a **less()** method that compares two students by their group numbers.
- Write a test program that prints and compares Student objects.

This is the end of CPT204-2425 Tutorial/Lab 1 Task Sheet.