CS2401 – Week 6-7

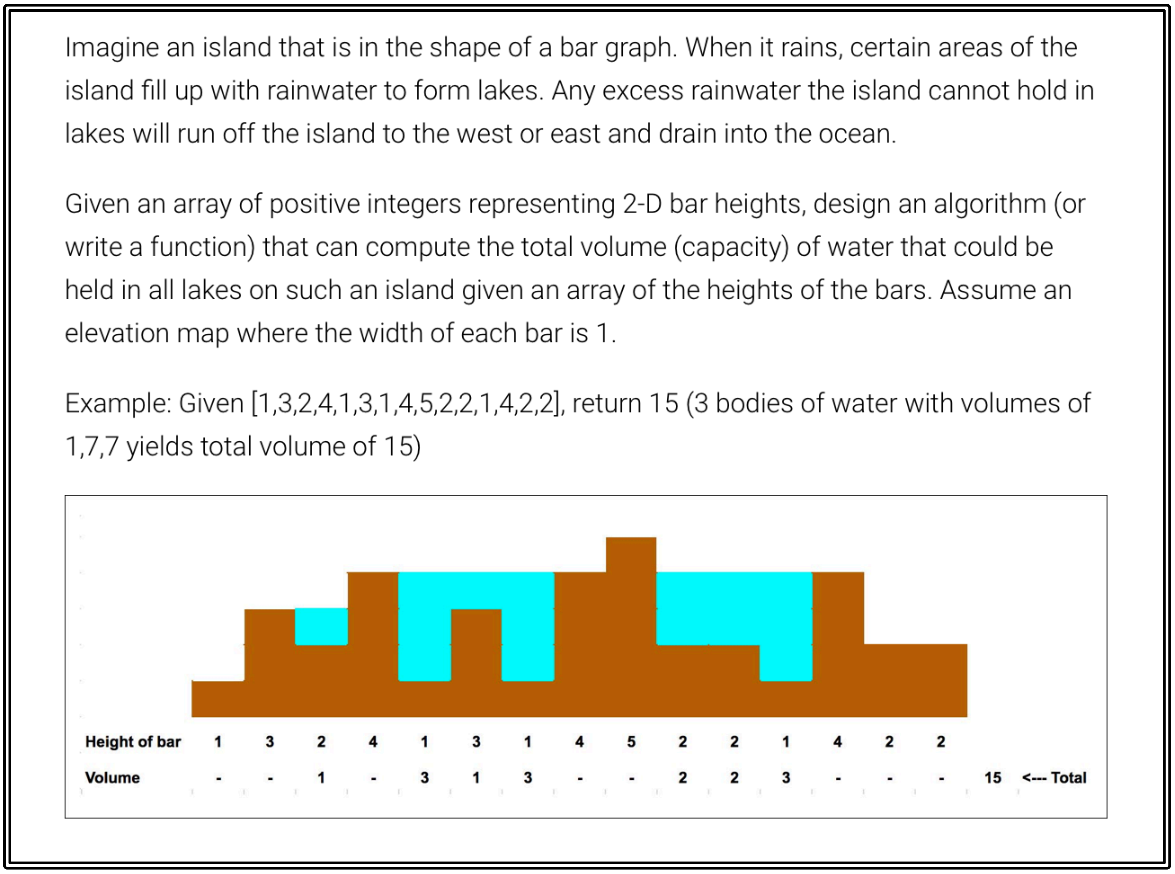
With this lab assignment, you are going to get a chance to practice some more on algorithms and time complexity. This lab assignment should be reasonably straightforward. We need you to read it very carefully before you start doing anything. You should be able to rephrase the instructions and split the lab into smaller units before you start. Let's get started!

What is the goal of this lab?

We expect that, by the time you complete this lab you will:

- Be more confident about designing and analyzing / testing algorithms;
- Better understand the concept of time complexity; and
- Complete a lab without too much tension / stress.

What problem will you be addressing in this lab? In this lab, you are going to write code to solve the following problem:



Imagine an island that is in the shape of a bar graph. When it rains, certain areas of the island fill up with rainwater to form lakes. Any excess rainwater the island cannot hold in lakes will run off the island to the west or east and drain into the ocean.

Given an array of positive integers representing 2-D bar heights, design an algorithm (or write a function) that can compute the total volume (capacity) of water that could be held in all lakes on such an island given an array of the heights of the bars. Assume an elevation map where the width of each bar is 1.

Example: Given [1,3,2,4,1,3,1,4,5,2,2,1,4,2,2], return 15 (3 bodies of water with volumes of 1,7,7 yields total volume of 15)

Height of bar	1	3	2	4	1	3	1	4	5	2	2	1	4	2	2		
Volume	-	-	1	-	3	1	3	-	-	2	2	3	-	-	-	15	<--- Total

[Source: https://techdevguide.withgoogle.com/resources/volume-of-water/#]

What do you have to do?

1/ Write a method to solve any instance of the above problem.
Your method called islandWater should take an array of positive integers and return an integer that corresponds to the amount of units of water that can fit on your island profile.
2/ Test this method on at least 5 test cases using JUnit test cases. For this, you will create a file called IslandTester.java. You are expected to describe each of the test cases in java comments right before each test case's code.
3/ Analyze the performance of your method islandWater. You are expected to conduct both a theoretical analysis and experimental study to determine the performance of your algorithm. The theoretical analysis has to be fully and clearly described in a word document. The code needed to conduct your experimental study needs to be added to your Island.java file. The experimental procedure needs to be described fully and clearly in a word document, and its results clearly reported in the same word document. Of utmost importance is that you explain why your experimental and theoretical results may not match.

What should you turn in?

You should submit 1/ Island.java, 2/ IslandTester.java, 3/ a word document called YourLastName-YourFirstName-Week6-7.docx in which you describe the performance of your method based on both a theoretical analysis and an experimental study of your method islandWater.

How should you submit your work?

You should **follow the specific submission instructions and guidelines given by your own TA** in lab.
Failing to follow submission instructions and guidelines given by your respective TA will result in up to 15 points off your overall grade in this lab. So please pay attention.
Additionally, your **word** and **java** files are expected to be neat and clear (organization, grammar, and spelling for the word file / indentation and clear variable naming for the java file). Failing to do so will result in up to 15 points off. On the other hand, extra neat and clear work will be rewarded by up to 10 extra points.

By when should you submit your work?

Due date: Saturday October 12 at 5 pm
Lateness rule: -10 pts for 1 day of lateness / - 20 pts for 2 days of lateness / 0 after that, but you still have to turn in your work

---

Grading:

20 pts	Method islandWater
25 pts	JUnit tests for method islandWater in IslandTester.java
10 pts	Theoretical analysis of performance
45 pts	Experimental analysis of performance: including additional code and testing strategy + report on these in word document