INSTITUTE OF TECHNOLOGY

TCSS 343 – Programming Assignment

August 2, 2018

1 GUIDELINES

This is a group assignment. Each group may consist of 2 or 3 students. Please state the names of your group members on your submission.

Details: In this assignment, you will implement algorithmic techniques in **Java**.

Homework should be electronically submitted via Canvas by midnight on the due date. Each group is expected to submit the following files:

- Report. The submitted report MUST be typeset using any common software and submitted as a PDF. We strongly recommend using LaTeX to prepare your solution. You could use any LaTeX tools such as Overleaf, ShareLatex, TexShop etc. When presenting your results, use log-scale graphics if/when results are too different (tiny vs. huge) to coexist on the same linear-scale graphic.
- Java source code. Submit one single Java file containing all your code, except for the challenge which must be in one single separate Java file.

  You must name your source code **tcss343.java** for the normal part of this assignment, and **challenge.java** for the bonus challenge.
- Your input testing files. Submit all the input testing files you documented in your report. The input testing files are tab-delimited text file. An example is provided on Canvas.

Execution. We will use the following command to execute your program for the normal part of this assignment:

```
java tcss343 < input.txt
```

The challenge has its own format describes below.
Remember to cite all sources you use other than the text, course material or your notes.

2 Problem Statement

There are n trading posts numbered 1 to n, as you travel downstream. At any trading post i, you can rent a canoe to be returned at any of the downstream trading posts $j > i$. You are given a cost array $R(i, j)$ giving the cost of these rentals for $1 \le i \le j \le n$. We will have to assume that $R(i, i) = 0$ and $R(i, j) = \infty$ if $i > j$. For example, with $n = 4$, the cost array may look as follows: The rows are the sources (i-s) and the columns are the destinations (j's).

	1	2	3	4
1	0	2	3	7
2		0	2	4
3			0	2
4				0

The problem is to find a solution that computes the cheapest sequence of rentals taking you from post 1 all the way down to post n. In this example, the cheapest sequence is to rent from post 1 to post 3 (cost 3), then from post 3 to post 4 (cost 2), with a total cost of 5 (less than the direct rental from post 1 to post 7, which would cost 7).

3 Your Tasks

3.1 Brute Force

(**10 points**): Design a brute force solution to solve this problem. You need to print the cheapest solution, as well as the sequence.

What is the asymptotic complexity of this algorithm?

3.2 Divide and Conquer

(**25 points**): Express the problem with a purely divide-and-conquer approach. Implement a recursive algorithm for the problem. Be sure to consider all sub-instances needed to compute a solution to the full input instance in the self-reduction, especially if it contains overlaps. As before, you need to print the solution, as well as the sequence.

What is the asymptotic complexity of this algorithm?

3.3 Dynamic Programming

(**25 points**): Design a dynamic programming table for this problem. How is the table initialized? In what order will the table be filled? How would you use the table to find the cheapest sequence of canoe rentals from post 1 to post n? Implement the corresponding dynamic programming solution.

What is the asymptotic complexity of this algorithm?

3.4 DOCUMENTATION

(**10 points**): Provide a well prepared document that describes your solutions, complexity analyses, and result analyses. Use log-scale graphics whenever the discrepancy between the cases you consider is so large than linear-scale graphics will make it too hard to compare those cases visually.

You must submit your code which should be well documented as well. If there is any known error in the code, you must point that out.

Also, document the division of labor (who did what) in your report.

3.5 TESTING

(**15 points**): Call a pseudo-random number generator to create a cost table of size $n \times n$ (notice only the upper diagonal of this matrix is full) for the following values of n: 25, 50, 100, 200, 400, 800.

For each n, consider two circumstances: in the first, the costs are entirely random (apart from being positive); in the second, the costs are random but increasing along each row of the cost matrix (add a random non-negative increment from the previous value on that row).

Save your input testing files as tab-delimited text files. An example input file corresponding to the example input described in Section 2 is provided on Canvas as ⟨`sample_input.txt`⟩. Include your java code of the above input generator in the source code of your submission.

Your code is expected to read in one input testing file, and generate the solution corresponding to **each of the three** approaches described in Sections 3.1, 3.2 and 3.3.

3.6 ANALYSIS OF RESULTS

(**15 points**): Analyze the results using visualization (you may use MS Excel or equivalent spreadsheet software for that). Use log-scale whenever the numbers you get are too discrepant to be easily distinguished in linear scale.

1. quality of the solutions (y-axis must present the cost, i.e., the objective function value) for varying x.
2. running time of the algorithms (y-axis will present the machine time the algorithms need to compute) for varying x.

Your analyses should be included in the submitted document.

Remember: use log-scale graphics if/when results are too different to coexist (i.e. when one results is tiny and the other is huge), both in the submitted documentation and in your final presentation.