CSE531: Algorithm Analysis and Design Fall 2024

Programming Project 2

Instructor: Kelin Luo **Deadline: Nov/18/2024**

Your Name: ______________________ Your Student ID: ________________

Problems	1	2	3	4	Total
Max. Score	10	10	10	10	40
Your Score					

Submission requirement:

- You should submit **each source file for each problem**. Name your file: Problem number_Last name_First name_YourStudents ID Number_ProjectNumber. Example: 1_Doe_John_55552222_P2
- Allowed programming languages: Python. Use Python version $\geq$ 3.4. You can run "python –version" to check the version. Please do not use any built-in functions unless they are explicitly stated in the provided code. Do not use any third-party packages other than the standard library.
- For each problem, please modify the provided file to include the algorithm (For example, `1_Doe_John_55552222_P2.py`). Note that the template code is just for the student to refer to get the correct input and output format, the student can revise the template based on their own code.
- The auto similarity-check will be applied for the programming submissions.
- The P2 deadline is 18 Nov, 11:59PM EST. Late submissions (within 24 hours with 25 % penalty or 48 hours with a 50% penalty) should be submitted via Email to Instructor and Head TAs. Submissions will close 48 hours after the deadline.
- Only the most recent submission is kept on Brightspace. Note that in each submission, please submit **all** your source files. Only files with extension .py will be accepted for the submission. You should view your submission after you upload it to make sure that it is not corrupted or malformed. You are responsible for making sure your submission went through successfully.
- The grading scheme and test cases will be released when the grades for Programming Project 2 are published.

Problem 1 (10 points). You need to implement the merge sort algorithm to find the k-th least populated city from a list of cities, where the ordering is based on the population of the cities. An expected $O(n \log n)$-time algorithm is sufficient to pass any feasible test cases.

Hint: use the merge sort algorithm to sort the cities and return the k-th least populated city.

Input You need to read the input from the console. The input consists of $n + 1$ lines of data. In the first line of the input, there are two positive integers n and k, where n represents the number of cities and k (with $1 \leq k \leq n$) denotes the rank of the city we would like to find based on its population. In the next n lines, each line i contains two positive integers: a_i, which is the index of the city, and p_i, representing the population of the city. You can assume that $1 \leq a_i \leq 1000$ and $1000 \leq p_i \leq 1000000$.

Output You need to output to console. The output is the k-th smallest city index and its population.

Example: Below are examples of input and output:

Example input:	Example output:

12 5
2 10000
1 10000
8 20000
3 20000
6 30000
5 42000
9 86000
10 10000
7 20000
11 67000
12 100000
4 93000
	3 20000

Example input:	Example output:

10 5
2 10000
1 10000
8 80000
3 15000
6 30000
5 42000
9 86000
10 10000
7 50000
4 93000
	6 30000

Problem 2 (10 points). Background: You are given two DNA sequences, sequence1 with size n characters and sequence2 with m characters. DNA sequences are composed of the characters 'A', 'C', 'G', and 'T'. Each DNA sequence may contain two consecutive sequences: sequence1=sequence11+sequence12 and sequence2=sequence21+sequence22. For example, sequence1='AGACTACCG', sequence1 may contain two consecutive sequences sequence11='' and sequence12='AGACTACCG', or sequence1 may contain two sequences sequence11='A' and sequence12='GACTACCG', ... or sequence1 may contain two sequences sequence11='AGACTACCG' and sequence12=''.

Your task is to find two sequences, sequence12 and sequence22, such that: (1) The length of the longest common subsequence (LCS) of sequence12 and sequence22 is equal to the length of the LCS of sequence1 and sequence2. (2) The total number of characters in sequence12 and sequence22 is minimized. The output should be the length of the LCS and the total length of the two sequences, sequence12 and sequence22, that you have found.

An $O(nm)$-time algorithm is suffice to pass any feasible test cases.

Hint: use the dynamic programming algorithm for longest common subsequence problem.

(a) **Input** You need to read the input from the console. It contains two lines, each containing one sequence. In the first line of the input, we have n characters; and in the second line of the input, we have m characters. You can assume that $1 \leq n \leq 10000$ and $1 \leq m \leq 10000$.

(b) **Output** You need to output to the console. The first line is an integer indicating the length of the LCS between the given two sequences. The second line is the total length of the two sequences, sequence12 and sequence22.

Below are example for input and output:

Example input:	Example output:
AGACTTAA	7
ATGACTATTCA	19

Example input:	Example output:
ACTTAA	4
TGCATTCA	11

Problem 3 (10 points). Given are an undirected graph with n nodes (vertices) and m edges, each having weights. Your task is to find a minimum spanning forest containing no more than k trees. Note that $0 \leq k \leq n$.

An $O(m \log^2 n)$-time algorithm is sufficient to pass all test cases.

Hint: use Kruskal's algorithm.

(a) **Input** You need to read the input from the console. In the first line of the input, we have three positive integers n, m and k. n is the number of vertices in the graph, m is the number of edges in the graph and k is a positive number. The vertices are indexed from 1 to n. You can assume that $1 \leq n \leq 1000$, $1 \leq m \leq 100000$ and $1 \leq k \leq n$. In the next m lines, each line contains 3 integers: u, v and w, with $1 \leq u < v \leq n$ and $1 \leq w \leq 100000$. This indicates that there is an edge (u, v) of weight $w > 0$. You can also assume that the graph is connected.

(b) **Output** You need to output to the console. The output is an integer indicating the total weight of the minimum spanning forest that contains no more than k trees, i.e., the minimum spanning forest that contains $\leq k$ trees.

Below are example for input and output:

Example input:	Example output:

5 5 1
1 2 2
1 3 1
2 3 2
2 4 3
4 5 4
	10

Example input:	Example output:

5 5 2
1 2 2
1 3 1
2 3 2
2 4 3
4 5 4
	6

Problem 4 (10 points). Given are a directed graph with n nodes (vertices) and m edges, each having weights. Your task is to find the shortest path from vertex 2 to vertex n.

An $O(n^3)$-time algorithm is sufficient to pass all test cases.

(a) **Input** You need to read the input from the console. In the first line of the input, we have two positive integers n and m. n is the number of vertices in the graph and m is the number of edges in the graph. The vertices are indexed from 1 to n. You can assume that $2 \leq n \leq 1000$ and $1 \leq m \leq 5000$. In the next m lines, each line contains 3 integers: u, v and w, with $1 \leq u \leq n$, $1 \leq v \leq n$ and $-1000 \leq w \leq 1000$. This indicates that there is an edge (u, v) from vertex u to vertex v. You can assume that there is no negative cycle in the graph.

(b) **Output** You need to output to the console. The output is an integer indicating the total length of the shortest path from vertex 2 and vertex n. If there is no path from vertex 2 to vertex n in the graph, the output is INFINITY.

Below are example for input and output:

Example input:	Example output:

5 5
1 2 2
1 3 -1
2 3 2
2 4 3
3 5 4
	6

Example input:	Example output:

5 4
1 2 2
1 3 -1
2 3 2
5 4 1
	INFINITY