CS 372/469 – Spring 2022
Lab 4
Due: 04/17/2022 11:59 pm

For each of the following questions, write a successful running code in any programming language that you prefer. Your code should run without any errors for any *valid* input.

All problems are borrowed from GeeksForGeeks.

Question 1 (30 points):

The goal of this question is to implement the Fractional Knapsack problem. The more popular 0-1 Knapsack problem (https://en.wikipedia.org/wiki/Knapsack_problem) will be part of Lab 5.

Given weights and values of n items, we need to put these items in a knapsack of capacity W to get the maximum total value in the knapsack. In the Fractional Knapsack problem, we can break items for maximizing the total value of knapsack.

Source (Open at your own risk – solution is also on this page):
https://www.geeksforgeeks.org/fractional-knapsack-problem/

Input and Output Example (from the above source):
Input will include Items as (value, weight) pairs and Knapsack capacity W
{{60, 10}, {100, 20}, {120, 30}}, W = 50

Output:

```
Maximum possible value = 240
```

Items to take full: :Item 1, 2

Items to take partially: 0.66 Item 3

Here, Item 1 is the first item in the list, i.e. {60, 10}, and so on.

Your code should be able to traverse the above input format (e.g. *{{60, 10}, {100, 20}, {120, 30}}, W = 50*) **from a given text file**.
Your algorithm must have a time complexity of O(n log n)

Question 2 (35 points):

The goal of this question is to implement Kruskal's Minimum Spanning Tree algorithm. You have to use the code for detecting a cycle that you wrote for Lab 2 (aka do not blindly copy from the solutions).

Source (Open at your own risk – solution is also on this page):
https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Input and Output Example (modified from the above source):

Input: *n = 5, e = 9*
A -> B (4), B -> C (3), C -> B (1), B -> D (2), B -> E (3), C -> E (5), E -> D (1), A -> C (2), C -> D (4)
Output (this is the format of the Output --- not the actual output of this example):
The MST will include the following edges:
A -> B (4), B -> C (3), C -> B (1)
The total cost of the MST is 8.

Your code should be able to traverse the above input format (e.g. *A -> B (4), B -> C (3), C -> B (1), B -> D (2), B -> E (3), C -> E (5), E -> D (1), A -> C (2), C -> D (4)*) **from a given text file** and create edges and its weights in your data structure.

Your algorithm must have a time complexity of O(ElogE) or O(ElogV)

Question 3 (35 points):

The goal of this question is to get practice at job scheduling algorithms.

There are n servers. You are given 2 arrays, A1 and A2. Array A1 contains information about amount of requests for each server. Hence, each server i is currently processing A1[i] amount of requests. There is another array A2 in which A2[i] represents the number of incoming requests that are scheduled to server i. Reschedule the incoming requests in such a way that each server i

holds an equal amount of requests after rescheduling. An incoming request to server i can be rescheduled only to server i-1, i, i+1. If there is no such rescheduling possible then output -1 else print number of requests hold by each server after rescheduling.

Source (Open at your own risk – solution is also on this page):
https://www.geeksforgeeks.org/schedule-jobs-server-gets-equal-load/

Input and Output Example (from the above source):

```
Input : A1 = {6, 14, 21, 1}
        A2 = {15, 7, 10, 10}
Output : 21
A2(0) scheduled to a(0) --> a(0) = 21
A2(1) scheduled to a(1) --> a(1) = 21
A2(2) scheduled to a(3) --> a(3) = 11
A2(3) scheduled to a(3) --> a(3) = 21
A1(2) remains unchanged --> a(2) = 21
```

```
Input : A1 = {1, 2, 3}
        A2 = {1, 100, 3}
Output : -1
No rescheduling will result in equal requests.
```

Your algorithm must have a time complexity of O(n)

Submission Instructions: Put all your solutions in a properly commented file named *lab4_lastname_firstname.EXTENSION*, where EXTENSION = the appropriate extension for the programming language that you chose.