Introduction to Algorithms	**Prelim 1 for cz473**
CS 4820, Spring 2021	**8pm Thursday, March 18**

- This prelim consists of 5 questions, on a total of 3 pages.
- For questions with multiple parts, the point value of each part is given at the start of the part.
- This is an open-book, open-notes exam. Submit your solutions electronically, in PDF format, to the assignment "Prelim 1" on CMS. If you handwrite your solutions, the PDF file you submit should be a single file containing a scan of every page of your solutions.
- When writing a solution, identify which problem number you are solving. If the solution spans more than one page, write the problem number on each page.
- When you are asked to design and analyze a polynomial-time algorithm, you must present a full description of the algorithm, an analysis of its running time, and a proof of its correctness, unless the problem says that you are allowed to skip one of those parts.
- Your solution can refer to a fact from the textbook, lectures, or homework. If you do so, you are not required to repeat the proof of the fact that your solution depends upon.

  Similarly, if you solve a problem by reducing to an algorithm from the textbook, lectures, or homework, then you do not need to re-derive the running time or proof of correctness for the algorithm you are reducing to. However, you should say which algorithm you are reducing to, and you should correctly account for that algorithm's running time when determining the overall running time of your own algorithm.
- **The policy on blank or partially blank solutions**: The same policy as in problem sets. If you turn in a blank solution, or if your entire solution consists of the words "no solution" or "left blank", you will get 20% of the credit for the problem. This 20% rule also applies at the level of individual parts of your solution, but only if you explicitly write "I don't know…" sentences indicating the parts that you're leaving out.

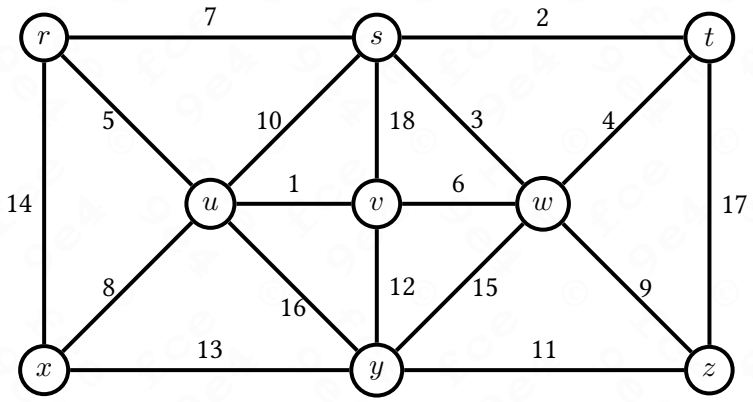
(1) *(6 points)* What is the running time of each of the following algorithms? Express your answers using big-O notation.

(1a) *(2 points)* The Gale-Shapley algorithm to compute a stable perfect matching of n firms and n workers.

(1b) *(2 points)* Kruskal's algorithm to compute a minimum spanning tree of a connected graph with n vertices and m edges.

(1c) *(2 points)* The Bellman-Ford algorithm to compute a minimum cost path in a graph with n vertices and m edges.

(2) *(8 points)* Consider the following instance of the minimum spanning tree problem. The graph has 18 edges whose costs are the numbers 1 through 18; in the diagram each edge is labeled with its cost.



(2a) *(4 points)* List the **costs** of the edges of the minimum spanning tree, in the order that **Prim's algorithm** starting from root node r would select them.

(2b) *(4 points)* List the costs of the edges of the minimum spanning tree, in the order that **Kruskal's algorithm** would select them.

(3) *(10 points)* One of the following statements about stable matchings is true, and the other is false. Prove one and give a counterexample to the other. (Correctly labeling a statement as true or false, without providing a proof or counterexample, will earn 40% partial credit.)

(3a) *(5 points)* If M is a stable perfect matching that matches every worker to the firm they rank *lowest*, then M must match every firm to the worker they rank *highest*.

(3b) *(5 points)* If M is a stable perfect matching that matches every firm to the worker they rank *highest*, then M must match every worker to the firm they rank *lowest*.

(4) *(12 points)* You are planning to order n items from a distributor. Each item i has a release date r_i and a deadline d_i, representing the earliest and latest deadlines, respectively, on which it may be ordered. (Assume that r_i and d_i are integers for each i.) You may order a set of items at any time t, subject to the constraint that item i must be ordered at a time t satisfying $r_i \leq t \leq d_i$. There is no limitation on the number of items in the set. The distributor charges a shipping cost for each order. This problem and the following one (problem 5) concern algorithms for minimizing the total shipping cost.

Suppose that the shipping cost of each order is a fixed constant, C. Consider the following two greedy algorithms for minimizing the total shipping cost to order the n items. Prove that one of them is correct, and find a counterexample for the other (i.e., an input instance that causes it to output an incorrect answer).

Correctly labeling an algorithm as correct or incorrect, without providing a proof or counterexample, will earn 40% of the credit. *You do not need to analyze the running time of these algorithms.*

Largest bundle first (LBF): Find a time t that maximizes the number of items in the set $S(t) = \{i \mid r_i \leq t \leq d_i\}$. Schedule an order for set $S(t)$ at time t. Delete the elements of $S(t)$ from the list of items, and recursively use the LBF algorithm to schedule orders for all of the remaining items.

Earliest deadline first (EDF): Let j be the item with the earliest deadline, let $t = d_j$, and let $R(t) = \{i \mid r_i \leq t \leq d_i\}$. Schedule an order for set $R(t)$ at time t. Delete the elements of $R(t)$ from the list of items, and recursively use the EDF algorithm to schedule orders for all of the remaining items.

(5) *(14 points)* As in problem 4, you are ordering n items, item i has release date r_i and deadline d_i and must be ordered at a time t satisfying $r_i \leq t \leq d_i$. However, in this problem the cost of ordering a set of items is defined differently. Suppose that each item has a shipping cost c_i, and that the cost of ordering set S is equal to

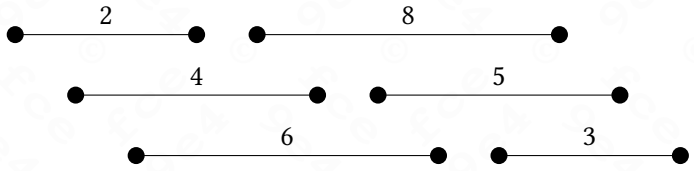
$$\text{cost}(S) = \min \left\{ C, \sum_{i \in S} c_i \right\}.$$

In other words, the cost of ordering a set of items is the sum of their shipping costs, up to a maximum cost of C, and for sets whose combined shipping cost exceeds this maximum the distributor charges only C.

(5a) *(4 points)* Consider an instance of the problem with six items having the following release dates, deadlines, and shipping costs. Define the maximum shipping cost, C, to be 10.

i	(r_i, d_i)	c_i
1	(1,4)	2
2	(2,6)	4
3	(3,8)	6

i	(r_i, d_i)	c_i
4	(5,10)	8
5	(7,11)	5
6	(9,12)	3



For $i = 1, 2, \ldots, 6$, what is the minimum cost of ordering items 1 through i? You don't need to show how you computed the answers, just write the answers in a table.

(5b) *(10 points)* Design an algorithm that solves the general case of this problem: your algorithm should compute the minimum total shipping cost for ordering the n items, subject to the release-time and deadline constraints. You can assume that the input specifies the items in order of increasing deadline, as they were in part (5a). Analyze your algorithm's running time and prove its correctness.