Basic Algorithms CSCI-UA.0310

Final Exam

Exam Time: 4:00 - 5:50 PM EST

You have to upload your solutions on Gradescope before 6:05 PM EST.

Instructions

- The exam is 110 minutes long. You have an additional 15 minutes to upload your solutions.
- **No submission after 6:05 PM EST will be accepted.** Please submit your solutions well ahead of time so that you will not encounter any connection issues.
- **You must answer each Problem on a separate page and must match your solutions to the corresponding problems while uploading them to Gradescope.**
- It is highly advised that you typeset your solutions via Latex, MS Word, etc. You can also scan your handwritten solutions but it is your responsibility to make sure that your solutions are clearly written and properly scanned. **Your solutions will NOT be considered if not clearly written.**
- You must stay connected to the Zoom meeting during the whole exam with your videos turned on.
- You may use any algorithm or theorem we saw in the class without proof as long as you state it correctly. For all questions asking to give algorithms, you have to fully explain your algorithm.
- Grading is based on correctness, clarity, and conciseness.

Requirements

- This exam is closed book. No calculators, phones, smartwatches, etc. are permitted.
- You may use your computer during the exam to only view this file, typeset your solutions, and upload your solutions.
- You may NOT use any communication tools to collaborate with others during the exam. This includes but is not limited to chat, messenger, email, etc.
- Anyone found sharing ideas, solutions, or communicating with another student during the exam period will earn an immediate grade of 0.

  You have to write the following statement at the beginning of your submission and sign it. Otherwise, your submission will not be considered.

  "I understand the ground rules and agree to abide by them. I will not share ideas, solutions, or assist another student during this exam, nor will I seek assistance from another student or attempt to view their ideas or solutions."

  Name:

  NYU Net ID:

Problem 0 (5 points)

Name your most favorite algorithm that was discussed in this course. You do NOT need to state the algorithm or explain it.

Problem 1 (10+10 points)

(a) Let $n \geq 3$ be a positive integer. Describe the point sets of size n on which the Jarvis' algorithm finds their convex hull faster than the Graham's algorithm. Justify your answer.

(b) Consider the undirected connected graph G that is also weighted. All the edge weights of G are distinct. Let T be the MST output by running the Prim's algorithm on G. We now decrease the weights of all edges of G by 1. For example, if the weight of an edge e was 20, then the new weight of e becomes $20 - 1 = 19$.
We run again the Prim's algorithm on G after decreasing all the edge weights by 1. Does the algorithm still output T? Justify your answer.

Hint: Try to draw an example for yourself.

Problem 2 (15 points)

You have n square-size papers and m square-size envelopes. Each letter has a length and each envelope has a length, as well. The lengths of the papers and the envelopes are given as input by the arrays $P[1 \ldots n]$ and $E[1 \ldots m]$, respectively, where $P[i]$ denotes the length of the ith paper, and $E[j]$ denotes the length of the jth envelope.
You can put a paper inside an envelope only if the length of the paper is less than the length of the envelope. You cannot put more than one paper in each envelope. Develop a greedy algorithm to put the maximum possible number of papers inside envelopes.
You must fully explain your algorithm. You do NOT need to write down the pseudo-code of your algorithm.

Problem 3 (10+10 points)

(a) Let $G = (V, E)$ be a directed graph with no cycles. As the input, we are also given the vertex s of G, and the positive integer k.
Develop an algorithm to determine if there is a path in G from s visiting at least k vertices of G. Your algorithm must run in $O(|V| + |E|)$ time.
You must fully explain your algorithm. You do NOT need to write down the pseudo-code of your algorithm.

(b) Let G be an undirected connected graph and let u, v be two distinct vertices of G. Bob runs BFS(G, v), and records the layer number of the vertex u appearing in the resulting BFS tree.
Now he runs DFS(G, v), and records the layer number of the vertex u appearing in the resulting DFS tree, as well.
After this experiment, he mentions that the layer number of the vertex u in the DFS tree is greater than the layer number of u in the BFS tree.
Note that the layer number of the root vertex v is always 0, the layer number of the vertices in the next layer is 1, and so on.
Do you think that Bob's answer is correct? Justify your answer.

Hint: Try to draw an example for yourself.

Problem 4 (20 points)

Sue has started to learn some graph theory. She is given a directed graph $G = (V, E)$. She labels the edges of G as *good edges* or *bad edges*. Thus, each edge of the graph G is labelled by her as either a *good edge* or a *bad edge*.
Now she picks two distinct vertices u, v of G, and wants to find the length of the path from u to v that has the least number of *bad edges*. In other words, among all the paths from u to v in G, she wants to find the length of the one that has the least number of *bad edges*.
Develop an algorithm that finds the length of such a path. Your algorithm must run in $O(|E| + |V| \log |V|)$ time.
Note that the length of a path is the number of edges that the path consists of.
You must fully explain your algorithm. You do NOT need to write down the pseudo-code of your algorithm.

Problem 5 (20 points)

Sue wants to explore more graph theory! She is again given a directed graph $G = (V, E)$, and she labels the edges of the graph as *good edges* or *bad edges*. Thus, each edge of the graph is labelled by her as either a *good edge* or a *bad edge*.
She picks two distinct vertices s, t of G, and wants to know if there is a walk from s to t using exactly an even number of *good edges*. Develop an $O(|V| + |E|)$-time algorithm to help Sue to determine if there is such a walk. Note that a walk may have repeated vertices.
You must fully explain your algorithm. You do NOT need to write down the pseudo-code of your algorithm.

Hint: Use an idea similar to the colored graph problems discussed in the homework and the practice test.