CSE101: Graphs test
Know your network

1 Instructions

1. You will get one question at 9AM, Monday March 16th. The question is provided as a pdf, sent to you over email.
2. The test is open book, but you must do the test yourself.
3. There will be a Zoom link for taking the test. You must come online and keep your camera on, so we can effectively proctor. You must be in view of the camera during the entire test.
4. You must submit through Canvas. Go to the "Assignments" link on the side. You will see an assignment for Test5.
5. To submit: write your solution in the text box provided.
6. Make sure you put down your name and UCSC email ID in your text box, just to be safe.
7. You will get 30 minutes for the test. **DRC students:** You will get 45 minutes or 60 minutes for the test, depending on your accommodation.
8. The question is an algorithms problem on graphs, though it might not explicitly talk about graphs. The problem will be closely related (but not identical) to some problem in the graphs questions posted on the website.
9. If the problem does not explicitly mention a graph, then you will likely have to construct in (in your algorithm). Clearly state what the vertices and edges of the graph are.
10. For your question, explain the algorithm using clearly written pseudocode. Some guidelines for pseudocode:
    (a) Number each line of pseudocode. Clearly demarcate loops with indentation. For clarity, also demarcate loops either using parentheses (like C/C++) or numbering with Roman numerals, letters, etc., to differentiate from the main pseudocode.
    (b) You do not need to write pseudocode for any algorithm/data structure taught in class. For example, you can write: "Run BFS on graph G to get the `pred` and `visited` arrays", and use the latter arrays. Or, you can "Run Dijkstra's shortest path algorithm on G with sources vertex s and weights (whatever)".
    (c) It's ok to get indexing wrong and have off-by-one errors.

11. For each question, you must give a running time analysis, in terms of big-Oh. For maximum clarity, it is best to write out the big-Oh running time of each Step of your algorithm (which is conveniently numbered!). If you have used some algorithm taught in class, you can simply state the running time. For example: "Mergesort runs in $O(n \log n)$, as explained in class."

12. Your running time will typically depend on n (the number of vertices) and m (the number of edges). If these quantities are not explicitly defined in your question, you will need to explain how they relate to the size of the input.

13. Try to use subscript-superscripts for math notation, so it is easier for us to read.

14. As long as your algorithm is correct and the running time analysis is correct, you will get full credit. You do not need to give an optimal (or the best) algorithm.

15. If there are some ambiguities, feel free to make reasonable decisions (but explain them!). For example, the question might not specify how to output (either print, or output as an object). Make reasonable choices, and as long as you get the algorithm correct, you will get credit. You are free to define objects that might be convenient, such as pairs or triples of integers. (You don't need to write out code for them.)