seminar 5 cmput 304 rbh fall 2020

1. Define the class P.
2. Give a problem in P.
3. Technically, SSSP is not in P. Why not?
4. Give a version of SSSP that is in P.
5. Define a yes-instance y of a decision problem D.
6. Give a yes-instance of 3-coloring.
7. Give a no-instance of 3-coloring.
8. Define the class NP.
9. Prove that 3-coloring is in NP.
10. Prove, disprove, or explain why you can neither prove nor disprove: every problem in P is in NP.
11. Prove, disprove, or explain why you can neither prove nor disprove: every problem in NP is in P.

1. The set of decision problems for which there is a worstcase polytime algorithm.
2. 2-coloring (given a graph, does it have a proper 2-coloring?). There are a gazillion other problems in P.
3. It's not a decision problem.
4. Given a graph G, two nodes x,y, and an integer k, is there a path from x to y with distance at most k?
5. An instance of D whose answer is yes.
6. A proper 3-coloring of a graph G is an assignment of at most 3 colors to nodes of G, so that every pair of adjacent nodes get different colors. 3-coloring is this problem: given a graph, does it have a proper 3-coloring? An example of a yes-instance of 3-coloring is any graph G with node set {a,b,c}.
7. An example of a no-instance of 3-coloring is the graph H with node set {a,b,c,d} and every pair of nodes adjacent.
8. The set of decision problems such that each yes-instance can be polytime verified.
9. 3-coloring is a decision problem. Next we have to prove that any yes-instance is polytime verifiable.

   So assume graph Y is a yes-instance of 3-coloring. So it has a proper 3-coloring (we don't have to find it, we just have to know that it exists): an assignment of colors 1,2,3 to the nodes, such that adjacent nodes get different colors. To verify that this is a proper 3-coloring, we only have to check that each node has color 1 or 2 or 3, and that each pair of adjacent nodes get different colors. We can do this by looking at each node once and each edge once, so runtime proportional to number of nodes plus edges, so polytime.
10. True. Use any polytime solver to verify any yes-instance.
11. We don't know.

    If every problem in NP is also in P, then NP is a subset of NP. We know from the previous question that P is a subset of NP, so we would have that P equals NP, and we would be famous and we would have won a millenium prize. But no one knows yet (or if they know, they aren't telling). whether P equals NP.

    If some problem in NP is not in P, then we would know that P does not equal NP, and again we would have won that millenium prize. But no one knows yet whether P equals NP.