Due at end of class: Friday, Oct 20, 2017

Name: ____________________________  Name: ____________________________

Name: ____________________________

Search & Sort worksheet

By Jim Xu

Please work in groups of 2 or 3 to work the following problems. Use additional paper as needed, and staple the sheets together before turning them in. ONLY TURN IN 1 WORKSHEET/ANSWERS PER TEAM.

1. [17] Below is an adjacency-matrix representation (denote as A) of graph G. (empty cell has value 0)

\[
\begin{array}{cccccc}
  & 1 & 2 & 3 & 4 & 5 & 6 \\
1 & & & & 1 & & \\
2 & & & & & & \\
3 & & & 1 & & & 1 \\
4 & & & 1 & & & 1 \\
5 & & & & & & \\
6 & & 1 & & & & \\
\end{array}
\]

\[a_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E, \\ 0 & \text{otherwise}. \end{cases}\]

A. [4] Graph G is a (weighted / unweighted) and (directed / undirected) graph. (circle correct choice)


1
2
3
4
5
6

C. [2] Which of the following represents a correct path? (circle correct choice)
   a. 1 -> 4 -> 2 -> 3
   b. 3 -> 5 -> 6 -> 3
   c. 6 -> 3 -> 3 -> 2
   d. 1 -> 4 -> 2 -> 2
D. [3] ______ vertex(s) has 0 out-degree, how can you find it(they) in adjacency-matrix?

E. [3] ______ vertex(s) has 0 in-degree, how can you find it(they) in adjacency-matrix?


2. [20] Based on the graph and answer questions below. While exploring, follow alphabetical order. Time cost for exploring one node to another node is, say, 1 unit.

A. [3] Directly write down the shortest path from r to x.

B. [6] Suppose the start point is q, use the Breath First Search (BFS) algorithm to find a path from q to q. (Use the code below if necessary)
   B1. [5] Write down each step when executing BFS, including the list(queue) you maintained and time cost for each node in the list(queue).
   B2. [1] Is this path a shortest path from q to q?
      Step1: $q$
      0

C. [5] Suppose the start point is r, use the Depth First Search (DFS) algorithm to find a path from r to w. (Use the code below if necessary)
   C1. [4] Write down all the steps as a list and the total time cost.
   C2. [1] Is this path a shortest path from r to w?
List: \[ r \rightarrow \]

D. [2] Comparing BFS and DFS, which one can lead to the shortest path from one node to another? Briefly explain why.

E. Play with code. (In the code below, s, v, u are vertexes.)

D1. [2] Change which line (one line) in BFS so that we can control the order of childNode we explore for each parentNode (say, exploring alphabetical order for each node). __________

D2. [2] After (below) which line (one line) in DFS can we insert a printNode() function so that it will print all the nodes we have explored in DFS graying order? ______________

D3. [2] After (below) which line (one line) in DFS can we insert a printNode() function so that it will print all the nodes we have explored in DFS blacking order? __________

BFS\( (G, s) \)

1. for each vertex \( u \in G.V - \{s\} \)
2. \( u.color = \text{WHITE} \)
3. \( u.d = \infty \)
4. \( u.\pi = \text{NIL} \)
5. \( s.color = \text{GRAY} \)
6. \( s.d = 0 \)
7. \( s.\pi = \text{NIL} \)
8. \( Q = \emptyset \)
9. ENQUEUE\( (Q, s) \)
10. while \( Q \neq \emptyset \)
11. \( u = \text{DEQUEUE}(Q) \)
12. for each \( v \in G.Adj[u] \)
13. if \( v.color == \text{WHITE} \)
14. \( v.color = \text{GRAY} \)
15. \( v.d = u.d + 1 \)
16. \( v.\pi = u \)
17. ENQUEUE\( (Q, v) \)
18. \( u.color = \text{BLACK} \)

DFS-VISIT\( (G, u) \)

1. \( time = time + 1 \)
2. \( u.d = time \)
3. \( u.color = \text{GRAY} \)
4. for each \( v \in G.Adj[u] \)
5. if \( v.color == \text{WHITE} \)
6. \( v.\pi = u \)
7. DFS-VISIT\( (G, v) \)
8. \( u.color = \text{BLACK} \)
9. \( time = time + 1 \)
10. \( u.f = time \)
3. [13] Below is a Prerequisite Chart of All Courses in Computer Science Major (From CSU CS website). Based on the Chart and answer the questions below. (View all types of line as required prerequisite)

A. [4] The Prerequisite Chart of All Courses in Computer Science Major above have valid topological sorts because it has __________ edges and __________.

B. [2] What courses are prerequisites of CS320 (consider one-level above)?
For the followings, (Assume we have studied CS163, CS164, all MATH and STAT courses, so we can view them as existed prerequisites)

C. [2] Write down the shortest series of courses you would take, ending in CS454, such that it respects the topological sort order (class prerequisites).

D. [3] Which of the following(s) is a valid topological sort? (circle all that apply)
   a. CS165, CS220, CS253, CS270, CS356
   b. CS220, CS270, CS253, CS314, CS453
   c. CS220, CS270, CS320, CS440, CS314
   d. CS165, CS220, CS270, CS370, CS320

F. [2] Play with code. Now we want to implement topological sort algorithm on this Chart in order to find a valid topological sort for all courses. Fill in the blank below to make it works as desired.

```python
1 - class course:
2    ""
3   Parameters
4     --------
5    self.inDegree : int
6       The number of edges comes into the course(vertex)
7    self.outCourses : List
8       A List of courses that dependent on this course
9    ""
10 -
11  def topoSor():
12     residentCourses = [All courses]
13     result = []
14 -
15     while residentCourses is not empty:
16 -
17         for course in residentCourses:
18 -
19             if __________________________:
20                 print("studying ", course)
21                 for dependentCourse in course.outCourses:
22                     dependentCourse.inDegree -= 1
23                 result.expand(course)
24                 residentCourses.remove(course)
25 -
26     print("a valid topoSor of the Graph is ", result)
```