CS317: Algorithms — Homework Assignment #7

Due: See Canvas for Assignment Due Dates (20 points)

NOTE: NO LATE ASSIGNMENTS WILL BE ACCEPTED because we often review them in class on the due date.

Please upload the document containing your answers. They can be handwritten and scanned, but they must be clearly legible to receive a grade on the assignment. PDF is the best format for canvas. You DO NOT Need to include this cover sheet in your upload. It is formatted for the grader to use for me, as needed.

Chapter 8: Work the following problems. Point values are provided for each problem.

Problem #	Points	Grader's Notes
Sec 8.1, #4	*2*	
Sec 8.2, #1	*3*	
Sec. 8.4, #1	*2*	
Sec. 8.4, #7	*2*	

Chapter 9: Work the following problems. Point values are provided for each problem.

Problem #	Points	Grader's Notes
Sec. 9.1, #7	*2*	
Sec. 9.1, #9b	*2*	
Sec. 9.2, #1b	*2*	
Sec. 9.3, #2b	*2*	

Chapter 12

Problem #	Point Value	Grader's Notes
Sec. 12.1, #1a	*2*	
Sec. 12.1, #7	*1*	

8.1

4. Apply the dynamic programming algorithm to find all the solutions to the change-making problem for the denominations 1, 3, 5 and the amount $n = 9$.

1	3	5
9	0	0

1	3	5
6	1	0

1	3	5
3	2	0

1	3	5
0	3	0

1	3	5
4	0	1

1	3	5
1	1	1

8.2

1. a. Apply the bottom-up dynamic programming algorithm to the following instance of the knapsack problem:

item	weight	value
1	3	$25
2	2	$20
3	1	$15
4	4	$40
5	5	$50

capacity $W = 6$.

b. How many different optimal subsets does the instance of part (a) have?

c. In general, how can we use the table generated by the dynamic programming algorithm to tell whether there is more than one optimal subset for the knapsack problem's instance?

Capacity

	i	0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
w=3, v=25	1	0	0	0	25	25	25	25
w=2, v=20	2	0	0	20	25	25	25	25
w=1, v=15	3	0	15	20	35	46	45	60
w=4, v=40	4	0	15	26	35	40	55	60
w=5, v=50	5	0	15	20	35	40	55	65

max = [5, 6];

optimal is item 3 & item 5.

b, c) If there is no equality between $V[i-1, j]$ and $v_i + V[i-1, j-w_i]$ then the instance of the knapsack problem has a unique and optimal solution.

8.4

1. Apply Warshall's algorithm to find the transitive closure of the digraph defined by the following adjacency matrix:

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R^{(0)} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad R^{(3)} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R^{(1)} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad R^{(4)} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R^{(2)} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

7. Solve the all-pairs shortest-path problem for the digraph with the following weight matrix:

D(0)

$$\begin{bmatrix} 0 & 2 & \infty & 1 & 8 \\ 6 & 0 & 3 & 2 & \infty \\ \infty & \infty & 0 & 4 & \infty \\ \infty & \infty & 2 & 0 & 3 \\ 3 & \infty & \infty & \infty & 0 \end{bmatrix}$$

D(1)

$$\begin{bmatrix} 0 & 2 & \infty & 1 & 8 \\ 6 & 0 & 3 & 2 & 14 \\ \infty & \infty & 0 & 4 & \infty \\ \infty & \infty & 2 & 0 & 3 \\ 3 & 5 & \infty & 4 & 0 \end{bmatrix}$$

D(2)

$$\begin{bmatrix} 0 & 2 & 5 & 1 & 8 \\ 6 & 0 & 3 & 2 & 14 \\ \infty & \infty & 0 & 4 & \infty \\ \infty & \infty & 2 & 0 & 3 \\ 3 & 5 & 8 & 4 & 0 \end{bmatrix}$$

D(3)

$$\begin{bmatrix} 0 & 2 & 5 & 1 & 8 \\ 6 & 0 & 3 & 2 & 14 \\ \infty & \infty & 0 & 4 & \infty \\ \infty & \infty & 2 & 0 & 3 \\ 3 & 5 & 8 & 4 & 0 \end{bmatrix}$$

D(4)

$$\begin{bmatrix} 0 & 2 & 3 & 1 & 4 \\ 6 & 0 & 3 & 2 & 5 \\ \infty & \infty & 0 & 4 & 7 \\ \infty & \infty & 2 & 0 & 3 \\ 3 & 5 & 6 & 4 & 0 \end{bmatrix}$$

D(5)

$$\begin{bmatrix} 0 & 2 & 3 & 1 & 4 \\ 6 & 0 & 3 & 2 & 5 \\ 10 & 12 & 0 & 4 & 7 \\ 6 & 8 & 2 & 0 & 3 \\ 3 & 5 & 6 & 4 & 0 \end{bmatrix}$$

$\boxed{9.1}$

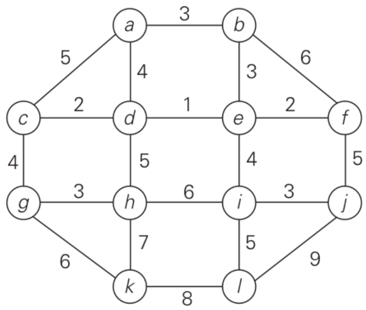
7. *Rumor spreading* There are n people, each in possession of a different rumor. They want to share all the rumors with each other by sending electronic messages. Assume that a sender includes all the rumors he or she knows at the time the message is sent and that a message may only have one addressee.

Design a greedy algorithm that always yields the minimum number of messages they need to send to guarantee that every one of them gets all the rumors.

Number students from 1 -> n as C1, C2, C3, ... Cn. Send n-1 messages C1->C2, C2->C3 ... Cn-1 ->Cn. Then send the message combining all the rumors from Student n -> C1, C2, ... Cn-1.

```
Greedy[
    Students C[ ];                      //Students
    while (! C[n-1])                    //while not at the last student
        Send C[i] -> C[i+1];            // send message to next student
        message = message + C[i+1]      //Store small message in large.
        i++;                            //increment
    Send message Student -> C[i]        // send whole message to all students.
]
```

9 **b.** Apply Prim's algorithm to the following graph. Include in the priority queue only the fringe vertices (the vertices not in the current tree which are adjacent to at least one tree vertex).

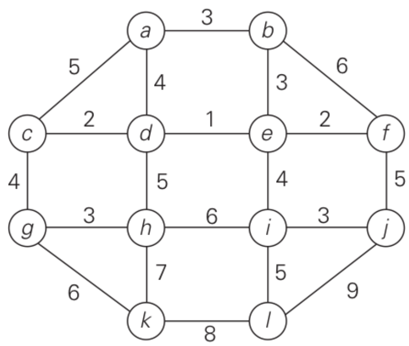


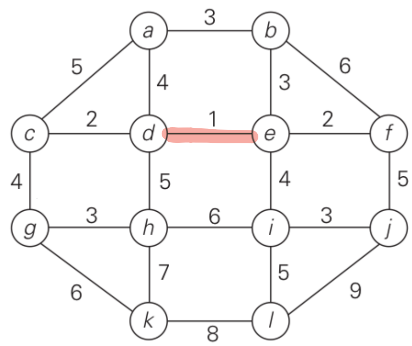
Tree vertices	Priority Queue
a (-, -)	b(a,3) c(a,5) d(a,4)
b (a,3)	c(a,5) d(a,4) e(b,3) f(b,6)
e (b,3)	c(a,5) d(e,1) f(e,2) i(e,4)
d (e,1)	c(d,2) f(e,2) i(e,4) h(d,5)
c (d,2)	f(e,2) i(e,4) h(d,5) g(c,4)
f (e,2)	i(e,4) h(d,5) g(c,4) j(i,5)
i (e,4)	h(d,5) g(c,4) j(i,3) l(i,5)
j (i,3)	h(d,5) g(c,4) l(i,5)
g (c,4)	h(g,3) l(i,5) k(g,6)
h (g,3)	l(i,5) k(g,6)
l (i,5)	k(g,6)
k (g,6)	

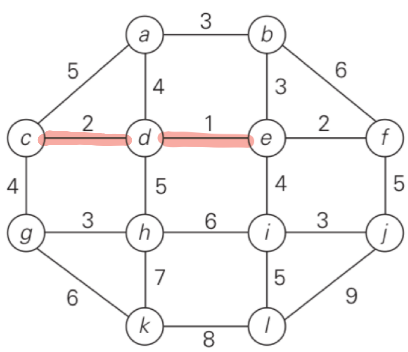
Minimum spanning tree w/ edges:

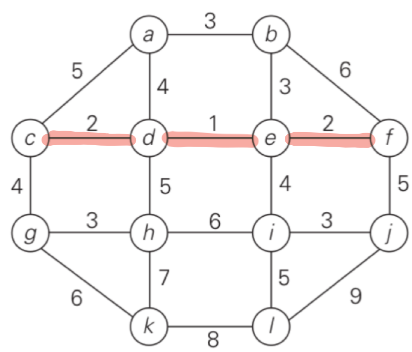
ab, be, ed, dc, ef, ei, ij, cg, gh, il, gk

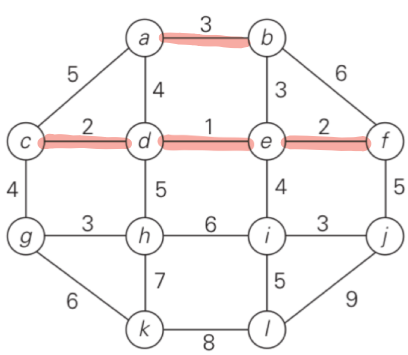
$\boxed{9.2}$ 1b)

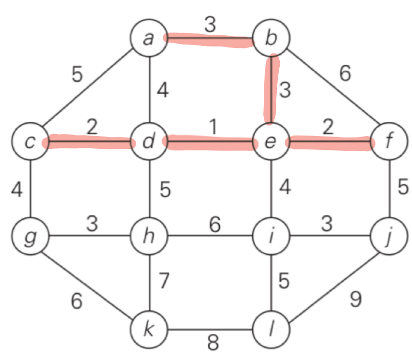


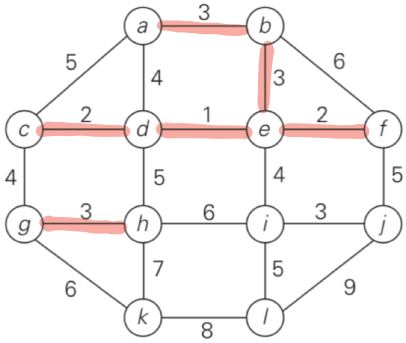


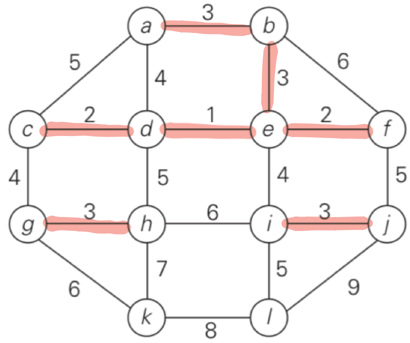


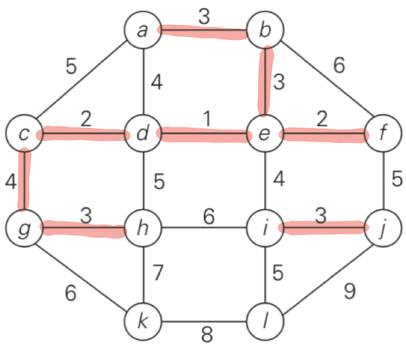


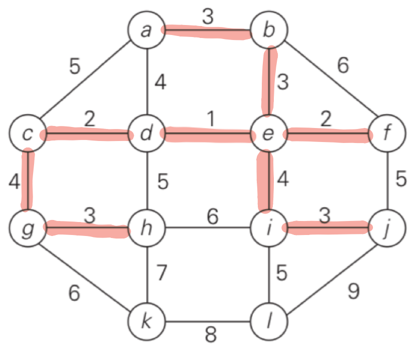


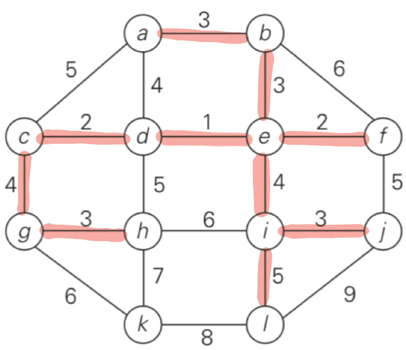


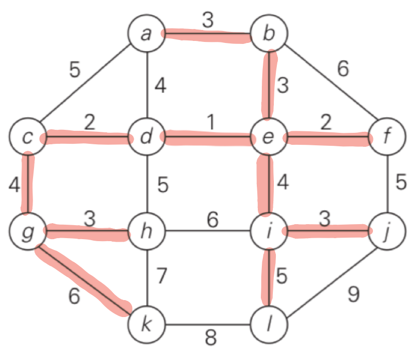

a 3 b
5 4 3 6
c 2 d 1 e 2 f
4 5 4 5
g 3 h 6 i 3 j
6 7 5 9
k 8 l


a 3 b
5 4 3 6
c 2 d 1 e 2 f
4 5 4 5
g 3 h 6 i 3 j
6 7 5 9
k 8 l

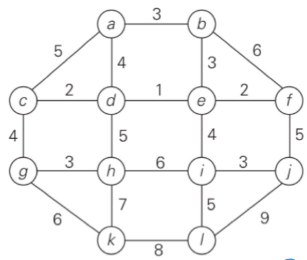

a 3 b
5 4 3 6
c 2 d 1 e 2 f
4 5 4 5
g 3 h 6 i 3 j
6 7 5 9
k 8 l


a 3 b
5 4 3 6
c 2 d 1 e 2 f
4 5 4 5
g 3 h 6 i 3 j
6 7 5 9
k 8 l


a 3 b
5 4 3 6
c 2 d 1 e 2 f
4 5 4 5
g 3 h 6 i 3 j
6 7 5 9
k 8 l


a 3 b
5 4 3 6
c 2 d 1 e 2 f
4 5 4 5
g 3 h 6 i 3 j
6 7 5 9
k 8 l

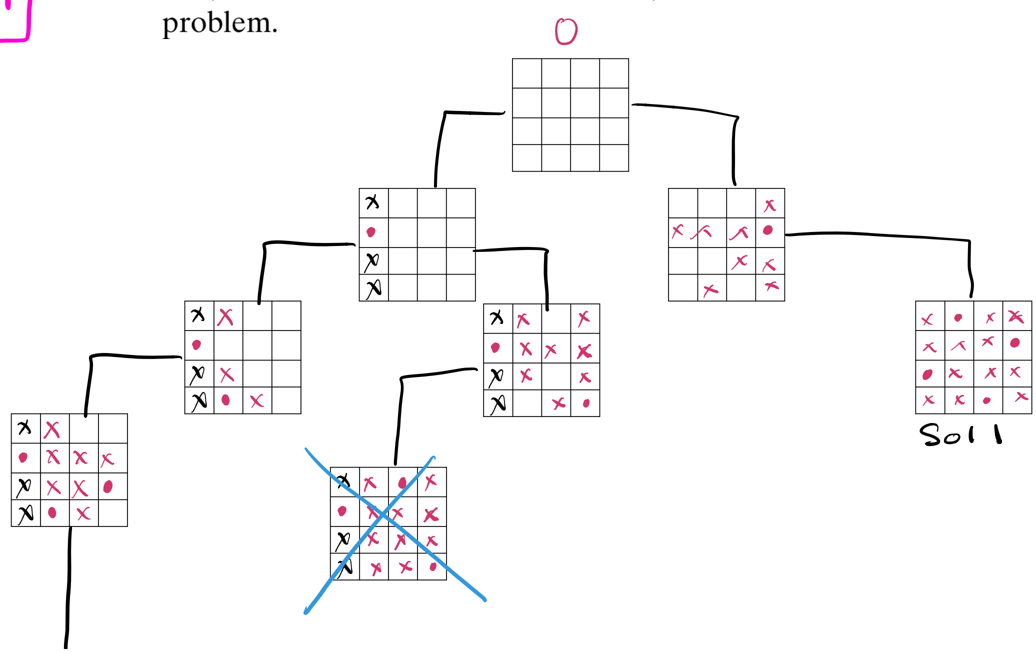
9.3 #2b

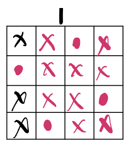


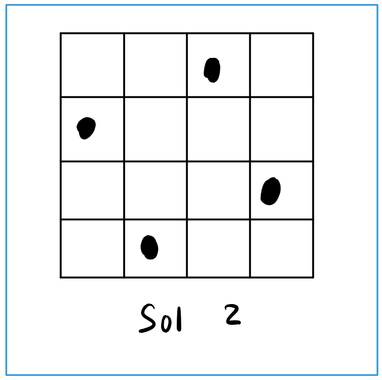
Tree vertices	Fring vertices	Shortest path from a.
a(-,0)	b(a,3) c(a,5) d(a,4)	to b: a-b L=3
b(a,3)	c(a,5) d(a,4) e(b,3+3) f(b,3+6)	to d: a-d L=4
d(a,4)	c(a,5) e(d,4+1) f(a,9) h(d,4+5)	to c: a-c L=5
c(a,5)	e(d,5) f(a,9) h(d,9) g(c,5+4)	to e: a-d-e L=5
e(d,5)	f(e,5+2) h(d,9) g(c,9) i(e,5+4)	to f: a-d-e-f L=7
f(e,7)	h(d,9) g(c,9) i(e,9) j(f,5+7)	to h: a-d-h L=9
h(d,9)	g(c,9) i(e,9) j(f,12) k(h,9+7)	to g: a-c-g L=9
g(c,9)	i(e,9) j(f,12) k(g,9+6)	to i: a-c-e-i L=9
i(e,9)	j(f,12) k(g,15) l(i,9+5)	to j: a-c-d-e-f-j L=12
j(f,12)	k(g,15) l(i,14)	to l: a-d-e-i-l L=14
l(i,14)	k(g,15)	to k: a-c-g-k L=15
k(g,15)		

12.1

1. a. Continue the backtracking search for a solution to the four-queens problem, which was started in this section, to find the second solution to the problem.







7. Generate all permutations of {1, 2, 3, 4} by backtracking.