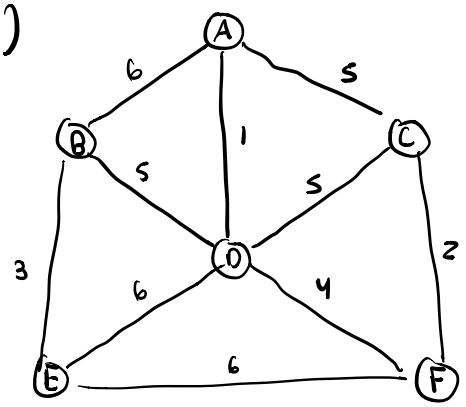
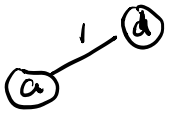
1)



a) Prim's

1 Choose A to add to set.

d(1,a) C(5,a) B(6,a) e(∞,a) F(∞,a)



2 Choose d to add to set

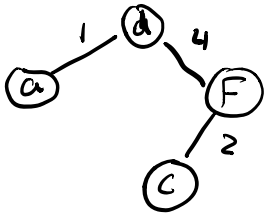
B(5,D), C(5,D) F(4,D) E(6,D)

3 Choose F to add to set

F(2,C) F(6,e)

4 Choose C to add to set

D(1,S) A(1,S)

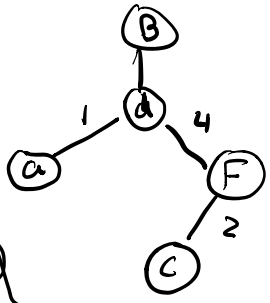


5 Choose D?

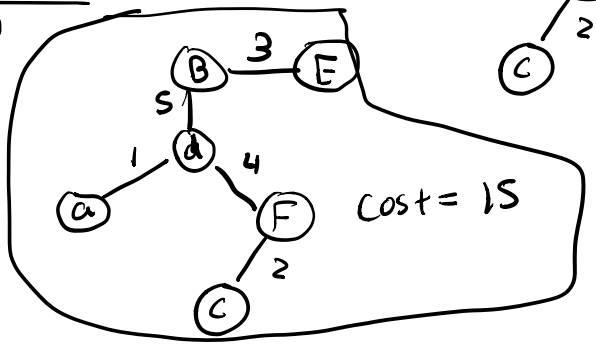
B(D,S) E(D,6)

6 Choose B

E(B,3) A(B,6) D(B,5)

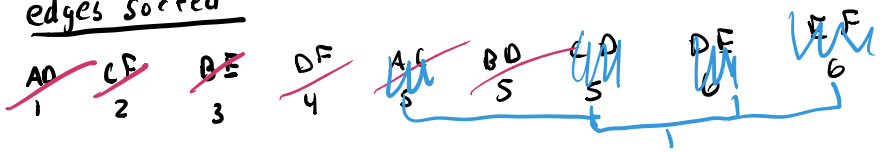


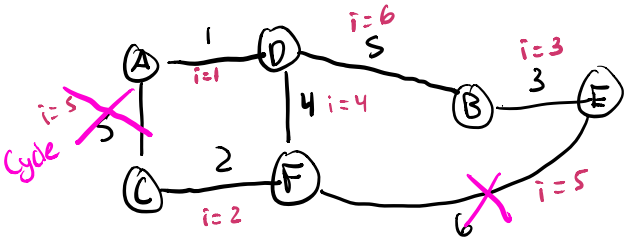
Choose E



b) Kruskal's

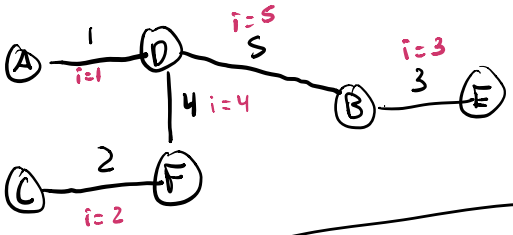
edges sorted





->note i=n is what iteration I added the connection

Final tree

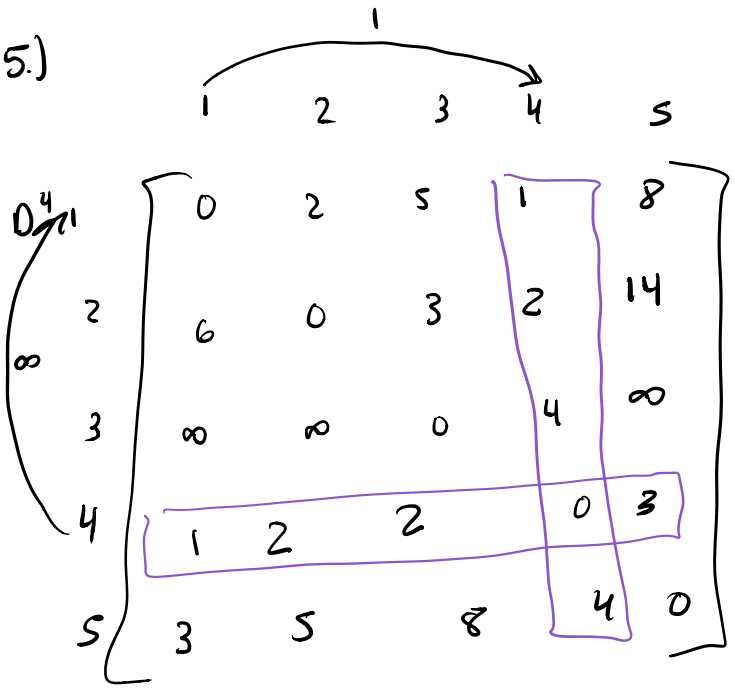


Cost= 1+5+3+2+4 = 15

3.) Dynamic programming works by solving an optimization problem by breaking down the problem into simpler sub problems and using the optimal sub problems to solve the overall problem in the most optimal solution. Overlapping sub problems occurs when the solution involves the same sub problem multiple times. Dynamic programming works so that if the sub problem is stored we do not have to recompute this when we see the same sub problem again.

4.) An optimal solution is one that works best for the given problem. I.e. where the objective solution reaches its maximum or minimum value. Usually there is not just one optimal solution. For example, the change making problem from #4 has two solutions where you have 3 coins. Three 3's or one 3, one 5, and one 1.

5.)

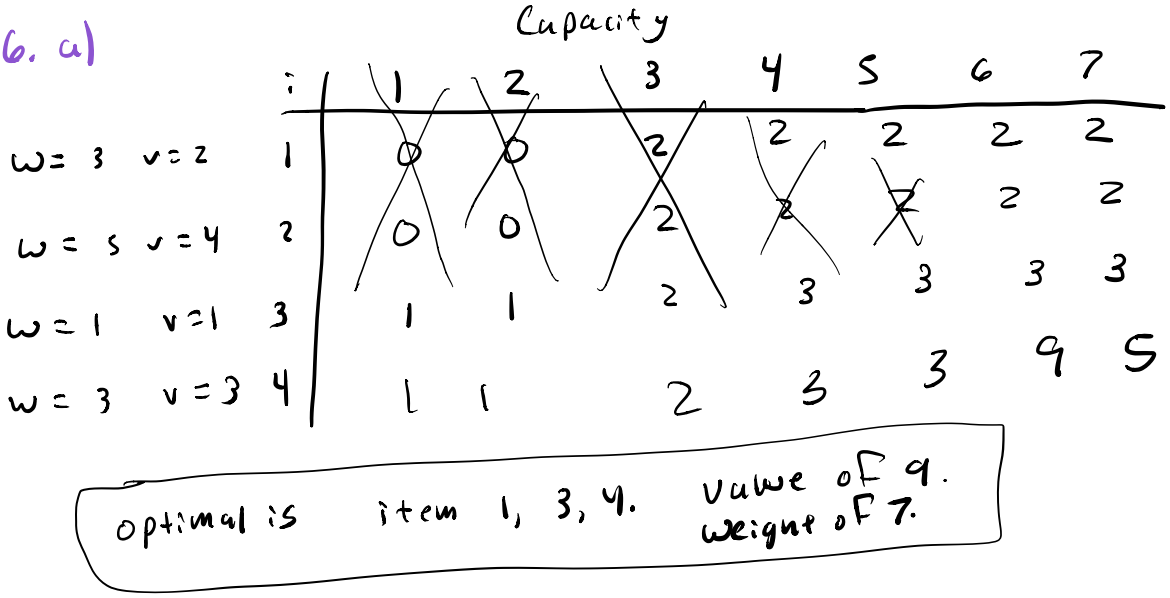


4->1 1->4 = 1

4,2 2,4 = 2

Not really sure on this one and running out of time.

6. a)

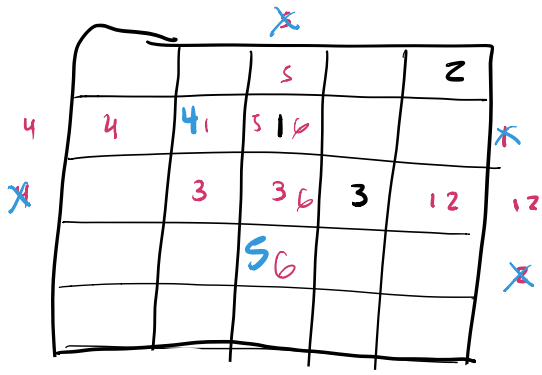


optimal is item 1, 3, 4. value of 9. weight of 7.

6. b). The greedy approach may fail by selecting 2 of item 1 (v=4, w=4) and 3 of item 3 which would make v=7, w=7. It is like this b/c it selects the first one and does not look at all of the values.

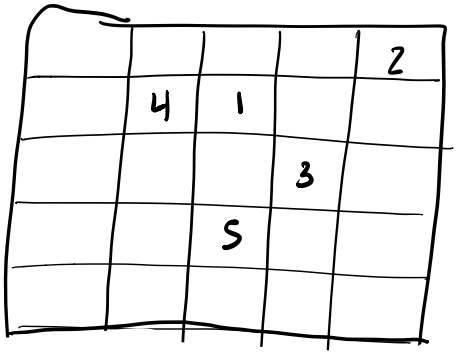
7) a) I am a little confused by the moves... does #1 start from 3? or is 1 on the board #1 in the list????

* I am assuming we start from 3 with #1 on the list since it is unclear.



1-> goes off board.

2-> goes off board.



* Check note above!

Final board with next ~~two~~ moves.

moves 3 & 6 are valid.

B.) if you run off the board or see that it does not work, you can backtrack to the previous one and start the next move.

C.) For N cells, it would need to test all of them and all of the moves that are capable for each one. There are N number different moves the knight can do. So it would most likely be NxN, but most likely more than that. I am running out of time.