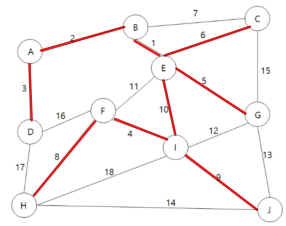
PA#4 Total 100 points

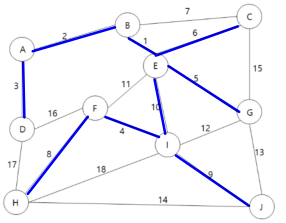
Topic	Minimum cost spanning tree & Shortest path [Total 54 points]

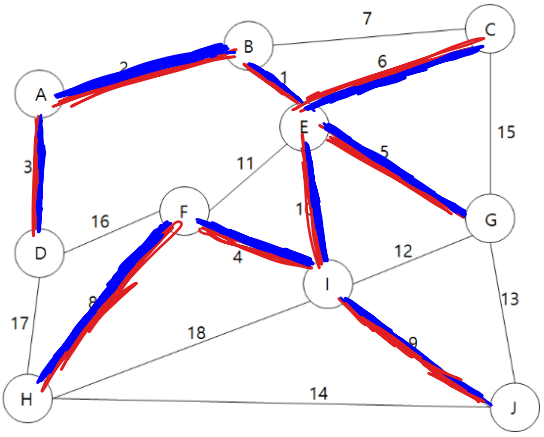
1. Find the Minimum Cost Spanning Tree of the graphs given below. Describe **in detail** the process of finding a spanning tree in each algorithm.

kruscal



prim





n=10

E = {(A,B,2), (B,C,7), (A,D,3), (B,E,1), (C,E,6), (E,F,11), (E,I,10), (E,G,5), (C,G,15), (D,F,16), (F,I,4), (I,G,12), (D,H,17), (F,H,8), (I,H,18), (I,J,9), (G,J,13), (H,J,14)}

A. Kruskal algorithm [**4 points** for minimum cost spanning tree **6 points** for description]

[Answer] T={ }

choose a least cost edge from E and delete from E

K_i = (V, W, cost) i : Number of executions

check T + K_i create a cycle → Yes ; discard (K_n)

No ; T + K_i ($K_1, K_2, K_3, K_4, K_5, K_6, K_8, K_9, K_{10}$)

K_1 = (B,E,1), K_2 = (A,B,2), K_3 = (A,D,3), K_4 = (F,I,4), K_5 = (E,G,5), K_6 = (C,E,6), K_7 = (B,C,7), K_8 = (F,H,8), K_9 = (I,J,9), K_{10} = (E,I,10)

T = {(B,E,1)^{K_1}, (A,B,2)^{K_2}, (A,D,3)^{K_3}, (F,I,4)^{K_4}, (E,G,5)^{K_5}, (C,E,6)^{K_6}, (F,H,8)^{K_8}, (I,J,9)^{K_9}, (E,I,10)$^{K_{10}}$}

if T contains (n-1) edges → break

if E is empty → break (NO MST)

B. Prim algorithm [**4 points** for minimum cost spanning tree **6 points** for description]

[Answer] T={ }, TV_0={A}

choose a least cost edge such that u ∈ TV_i and v ∉ TV_i or v ∈ TV_i and u ∉ TV_i

P_i = (u, v, cost) i : Number of executions

if there is no such edges → break

P_1 = (A,B,2), TV_1 = {A,B}

P_2 = (B,E,1), TV_2 = {A,B,E}

P_3 = (A,D,3), TV_3 = {A,B,E,D}

P_4 = (E,G,5), TV_4 = {A,B,E,D,G}

P_5 = (C,E,6), TV_5 = {A,B,E,D,G,C}

P_6 = (E,I,10), TV_6 = {A,B,E,D,G,C,I}

P_7 = (F,I,4) TV_7 = {A,B,E,D,G,C,I,F}

P_8 = (F,H,8) TV_8 = {A,B,E,D,G,C,I,F,H}

P_9 = (I,J,9) TV_9 = {A,B,E,D,G,C,I,F,H,J}

TV_i + P_i.V

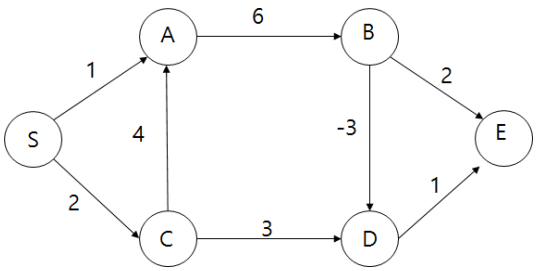
T + P_i

if T contains (n-1) edges → break

T = {(A,B,2)^{P_1}, (B,E,1)^{P_2}, (A,D,3)^{P_3}, (E,G,5)^{P_4}, (C,E,6)^{P_5}, (E,I,10)^{P_6}, (F,I,4)^{P_7}, (F,H,8)^{P_8}, (I,J,9)^{P_9}}

2. Solve the problem for the graph given below.

A. Apply the algorithm of Dijkstra to find the shortest path from S to D. Write down how the value of the data structure shown below changes in the process by which Algorithm operates. Each step carries 6 points.



[Answer]

distance: Save the cost of the shortest path found so far.

found: Save whether the shortest path has been determined.

Pi: store vertex id that reduced cost of shortest path.

<step 1>: Initialization [6 points]

distance					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	1	1000	2	1000	1000

found					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
1	0	0	0	0	0

pi					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	0	-1	0	-1	-1

<step 2> [6 points]

distance					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	1	7	2	1000	1000

found					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
1	1	0	0	0	0

pi					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	0	1	0	-1	-1

<step 3> [6 points]

distance					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	1	7	2	9	1000

found					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
1	1	0	1	0	0

pi					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	0	1	0	3	-1

<step 4> [6 points]

distance					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	1	7	2	5	6

found					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
1	1	0	1	1	0

pi					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	0	1	0	3	4

<step 5> [6 points]

distance					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	1	7	2	5	6

found					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
1	1	0	1	1	1

pi					
[0] S	[1] A	[2] B	[3] C	[4] D	[5] E
0	0	1	0	3	4

Answer of shortest path from S to D is: S → C → D [4 points] .

but, since the graph has an negative edge

, It is not the exact shortest path from S to D

Topic	Sorting [Total 46 points]

1. The descriptions of sorting algorithms given below *are wrong* in some way. Correct the sentences to give the correct descriptions. Explain your answer in detail. Assume that the number of records to be processed is n.

   A. Both Quick and Merge sort has a best-case time complexity of O(~~n~~ nlogn). [4 points]

   B. When using ~~selection~~ Insertion sort, if you insert one element in an already sorted array, this insertion is done with time complexity of O(n). [4 points]

   C. The method of selecting pivot in Quick sort ~~does not~~ affect time complexity. [4 points]
   If the pivot is badly selected (ex. maximum value) then quick sort have a big impact on time complexity

   D. The average ~~and worst case~~ scenario for quick sort is *n.**log**n.* [4 points]
   quick sort worst case : $O(n^2)$
   average case : $O(n \log n)$

2. We want to sort the following array using Quick sort. Pivot uses the leftmost record.

5	7	4	9	8	5	6	3

   A. Write down the contents of the array after the first split of quick sort. [2 points]

   [Answer] 5 3 4 5 8 9 6 7

   B. How many record changes occur in the process of doing the first split? [2 points]

   [Answer] 3 (7,3), (9,5), (5,5)

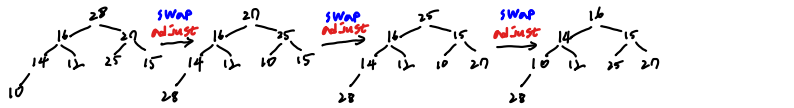
   C. After the first step of quick sort, elements on the left list are smaller than the pivot and the elements on the right list are larger than the pivot. Will the position of this pivot change or not when the next step proceeds? Explain your Answer? [4 points]

   [Answer] After the first step of quick sort, this pivot is aligned position. so it won't change

3. Let's say we sort an array with eight integers by applying Heapsort. In the middle of applying Heapsort, if the records in the array are as follows, how many adjustments have been performed at most? Write down the reason why you think so. [6 points]

index	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
data	16	14	15	10	12	25	27	28

[Answer]



at most 3 adjustment

4. Suppose that n records are entered in ascending order as follows. These records are sorted by applying the following sorting algorithm. For each sorting algorithm, express how time complexity is expected in asymptotic notification and write down the reason.

Records: 1 2 3 4 5 6 .... (n-2) (n-1) n

A) Quick sort [4 points]

B) Insertion sort [4 points]

C) Bubble Sort [4 points]

D) Heap sort [4 points]

[Answer]

these records are already sorted

(A) Quick sort
- Pivot; middle → O(nlogn) by split is exactly divided in half
- pivot; leftmost → O(n^2) by split is not efficiently achieved

(B) Insertion sort → O(n) by These records are already sorted, no additional work is required

(C) Bubble Sort → O(n^2) by Regardless of whether the records are sorted, all operations performed

(D) Heap Sort → O(nlogn) by Regardless of whether the records are sorted, all operations performed

Point deduction policy:

Late submission penalty: *minus 10 points.*