CS 300 HW4

1)

Either one of the two partitions with sizes k and N-K-1 will be quick selected or the k^{th} smallest element is found.

$$\mathrm{T(N)} = cN + \frac{k}{N}\mathrm{T(k)} + \frac{N-k-1}{N}T(N-k-1) + \frac{1}{N}0$$

The average complexity of the recursive call is replaced.

$$\mathrm{T(N)} = cN + 1/N\sum_{k=0}^{N-1}\frac{2k}{N}\mathrm{T(k)} = \mathrm{T(N)} = cN + 1/N^2\sum_{k=0}^{N-1}2k\mathrm{T(k)}$$

Both sides are multiplied with N^2.

$$N^2\mathrm{T(N)} = cN^3 + \sum_{k=0}^{N-1}2k\mathrm{T(k)}$$

N is replaced with N-1.

$$(N-1)^2\mathrm{T(N-1)} = c(N-1)^3 + \sum_{k=0}^{N-2}2k\mathrm{T(k)}$$

Do subtraction.

$$N^2\mathrm{T(N)} - (N-1)^2\mathrm{T(N-1)} = c(N^3-(N-1)^3) + 2(\mathrm{N}-1)\mathrm{T(N-1)}$$

Then,

$$N^2\mathrm{T(N)} = c(N^3-(N-1)^3) + (N^2-2\mathrm{N}+1+2\mathrm{N}-2)\mathrm{T(N-1)} = c(3N^2-3N+1) + (N^2-1)\mathrm{T(N-1)}$$

The equation is divided by N(N+1).

$$\frac{NT(N)}{N+1} = c\left(\frac{3(N-1)}{N+1} + \frac{1}{N} - \frac{1}{N+1}\right) + \frac{N-1}{N}T(N-1)$$

Let cf(N) be $\frac{NT(N)}{N+1}$. Then,

f(N) $= \left(\frac{3(N-1)}{N+1} + \frac{1}{N} - \frac{1}{N+1}\right) + f(N-1)$

$$= \sum_{k=1}^{N} (\frac{3(k-1)}{k+1} + \frac{3(k-1)}{k+1} - \frac{3(k-1)}{k+1})$$

$$= \sum_{k=1}^{N} (\frac{3(k+1)-6}{k+1} + \frac{1}{k(k+1)})$$

$$= \frac{N}{N+1} + 3N - 6\left(\sum_{k=1}^{N} \frac{1}{k+1}\right)$$

Finally,

f(N) = $\frac{N}{N+1} + 3N - 6\ln(N)$

Since f(N) = $\frac{NT(N)}{c(N+1)}$,

$\frac{NT(N)}{c(N+1)}$ = $\frac{N}{N+1} + 3N - 6\ln(N)$

Then,

T(N) = c $+ 3c(N+1) - \frac{6c(N+1)\ln(N)}{N}$

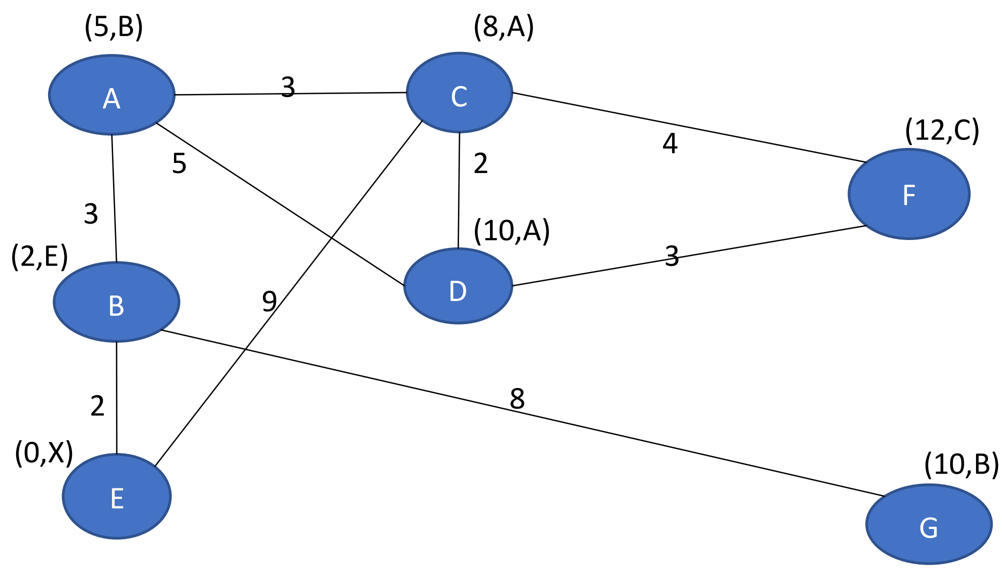
Thus, complexity of T(N) is O(N).

2) At the beginning, all of the node's status are UNKNOWN and their distances are ∞

Below are the updates at each iteration:

1. E.status = KNOWN, E.distance = 0 ,B.distance = 2, C.distance = 9, E.path = X(the first node), B.path = E, C.path = E
2. B.status = KNOWN, A.distance = 3+2=5, G.distance = 8+2=10, A.path = B, G.path = B
3. A.status = KNOWN, C.distance = 5+3=8, D.distance = 5+5=10, C.path = A, D.path = A
4. C.status = KNOWN, F.distance = 8+4=12, F.path = C
5. G.status = KNOWN
6. D.status = KNOWN
7. F.status = KNOWN

At the end:
A(B,5), B(E,2), C(A,8), D(A,10), E(E,0), F(C,12), G(B,10)



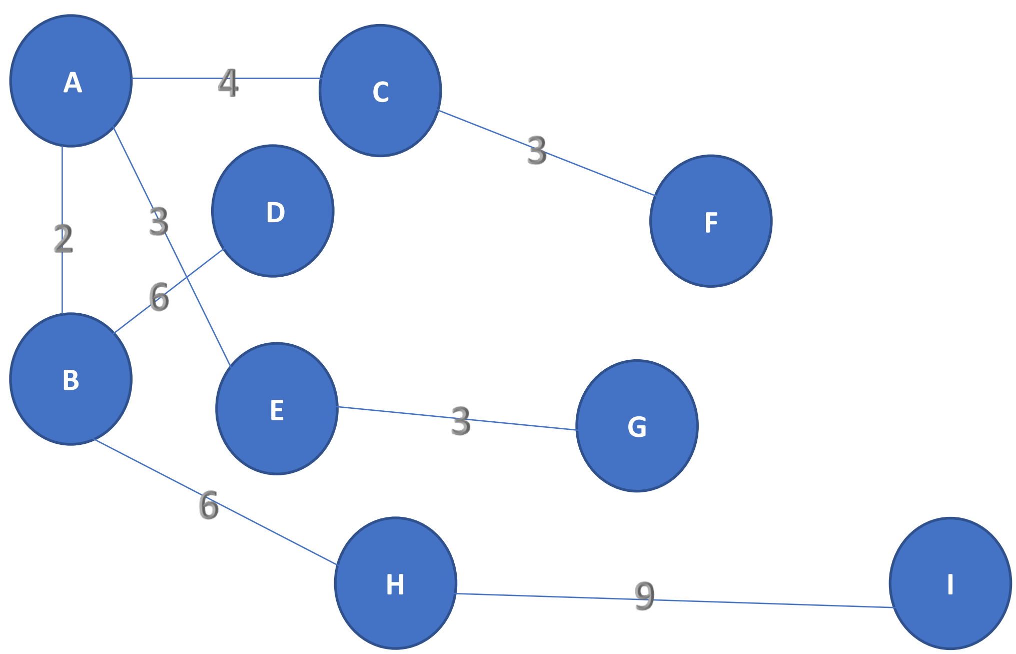
3) At the Beginning: Vertices in the tree = -
Edges in the tree = -

1. Vertices = E / Edges = - / Possible edges = EA(3), ED(8), EG(3)
2. Vertices = E, A / Edges = EA(3) / Possible edges = ED(8), EG(3), AB(2), AC(4)
3. Vertices = E, A, B / Edges = EA(3), AB(2) / Possible edges = ED(8), EG(3), AC(4), BH(6), BD(6)

4. Vertices = E, A, B, G / Edges = EA(3), AB(2), EG(3) / Possible edges = ED(8), AC(4), BH(6), BD(6), GF(5), GI(10)
5. Vertices = E, A, B, G, C / Edges = EA(3), AB(2), EG(3), AC(4) / Possible edges = ED(8), BH(6), BD(6), GF(5), GI(10), CD(8), CH(6), CF(3)
6. Vertices = E, A, B, G, C, F / Edges = EA(3), AB(2), EG(3), AC(4), CF(3) / Possible edges = ED(8), BH(6), BD(6), GI(10), CD(8), CH(6) , BD(6)
7. Vertices = E, A, B, G, C, F, H / Edges = EA(3), AB(2), EG(3), AC(4), CF(3), BH(6) / Possible edges = ED(8), BD(6), GI(10), CD(8), HI(9)
8. Vertices = E, A, B, G, C, F, H, D / Edges = EA(3), AB(2), EG(3), AC(4), CF(3), BH(6), BD(6) / Possible edges = GI(10), HI(9)
9. Vertices = E, A, B, G, C, F, H, D, I / Edges = EA(3), AB(2), EG(3), AC(4), CF(3), BH(6), BD(6), HI(9) / Possible edges = -

The iterations end since there are 8 = N-1 edges.
TOTAL DISTANCE = 3 + 2 + 3 + 4 + 3 + 6 + 6 + 9 = 36

The minimum spanning tree is as follows.



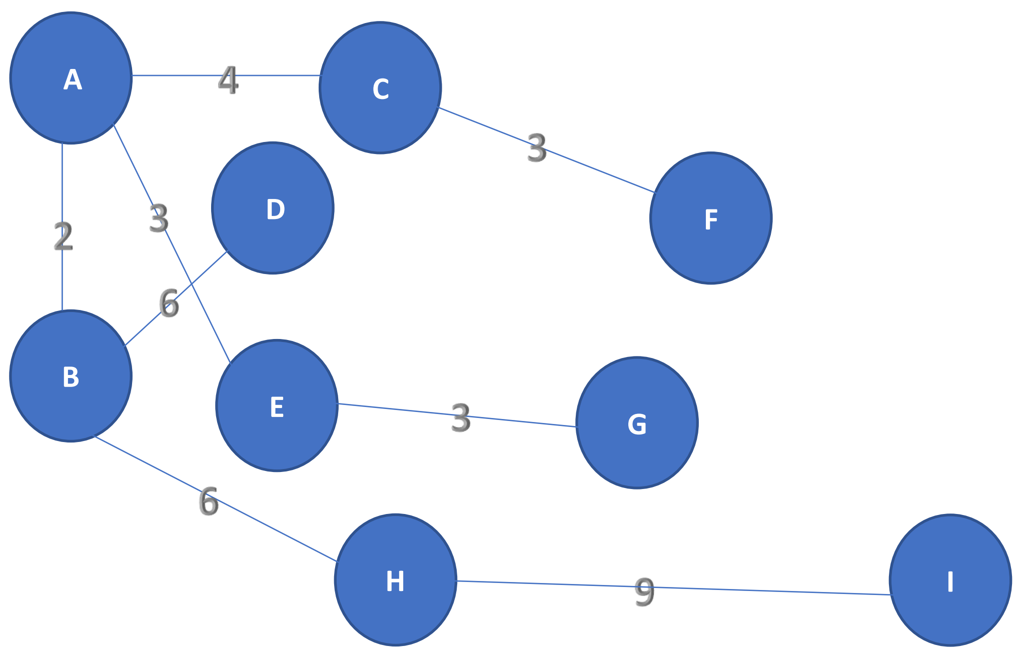
4) At the beginning: Edges in the tree = -

1. Edges = - / Possible edges = AB(2), AE(3), EG(3), CF(3), AC(4), GF(5), BD(6), BH(6), CH(6), DC(8), DE(8), HI(9), GI(10)
2. Edges = AB(2) / Possible edges = AE(3), EG(3), CF(3), AC(4), GF(5), BD(6), BH(6), CH(6), DC(8), DE(8), HI(9), GI(10)

3. Edges = AB(2), AE(3) / Possible edges = EG(3), CF(3), AC(4), GF(5), BD(6), BH(6), CH(6), DC(8), DE(8), HI(9), GI(10)
4. Edges = AB(2), AE(3), EG(3) / Possible edges = CF(3), AC(4), GF(5), BD(6), BH(6), CH(6), DC(8), DE(8), HI(9), GI(10)
5. Edges = AB(2), AE(3), EG(3), CF(3) / Possible edges = AC(4), GF(5), BD(6), BH(6), CH(6), DC(8), DE(8), HI(9), GI(10)
6. Edges = AB(2), AE(3), EG(3), CF(3), AC(4) / Possible edges = BD(6), BH(6), CH(6), DC(8), DE(8), HI(9), GI(10) / Cycle Constructor: GF(5)
7. Edges = AB(2), AE(3), EG(3), CF(3), AC(4), BD(6) / Possible edges = BH(6), CH(6), HI(9), GI(10) / Cycle Constructor: GF(5), DC(8), DE(8)
8. Edges = AB(2), AE(3), EG(3), CF(3), AC(4), BD(6), BH(6) / Possible edges = HI(9), GI(10) / Cycle Constructor: GF(5), DC(8), DE(8), CH(6)
9. Edges = AB(2), AE(3), EG(3), CF(3), AC(4), BD(6), BH(6), HI(9) / Possible edges = - / Cycle Constructor: GF(5), DC(8), DE(8), CH(6), GI(10)

The iterations end since there are 8 = N-1 edges.
TOTAL DISTANCE = 2 + 3 + 3 + 3 + 4 + 6 + 6 + 9 = 36

The minimum spanning tree is as follows.



5) At the beginning, states of all of the vertices are UNVISITED and distances are ∞, and paths are ∞
Starting from G, we will visit all the neighbors of the next vertex with the distance ∞

1. G.dist = 0, G.path = X
2. G.state = VISITED, E.dist = 1, E.path = G, F.dist = 1, F.path = G, I.dist = 1, I.path = G
3. E.state = VISITED, A.dist = 2, A.path = E, D.dist = 2, D.path = E
4. F.state = VISITED, C.dist = 2, C.path = F
5. I.state = VISITED, H.dist = 2, H.path = I
6. A.state = VISITED, B.dist = 3, B.path = A
7. D.state = VISITED
8. C.state = VISITED
9. H.state = VISITED
10. B.state = VISITED

Iterations end since all of the vertices are VISITED
Shortest Paths:

A = G – E – A (2)
B = G – E – A – B (3)
C = G – F – C (2)
D = G – E – D (2)
E = G- E (1)
F = G – F (1)
G = G (0)
H = G – I – H (2)
I = G – I (1)

6) I picked one of the vertices with indegree 0 randomly. Then I printed it. Finally, I deleted that vertex and the edges associated with it.

1. Vertices with indegree 0: {S} / Picked S
2. Vertices with indegree 0: {B,A} / Picked A
3. Vertices with indegree 0: {B,D} / Picked B
4. Vertices with indegree 0: {D,C} / Picked D
5. Vertices with indegree 0: {C} / Picked C
6. Vertices with indegree 0: {F,G} / Picked G
7. Vertices with indegree 0: {E,F} / Picked E
8. Vertices with indegree 0: {F} / Picked F
9. Vertices with indegree 0: {T} / Picked T

S - A – B – D – C – G – E – F - T