EL9343 Homework 11

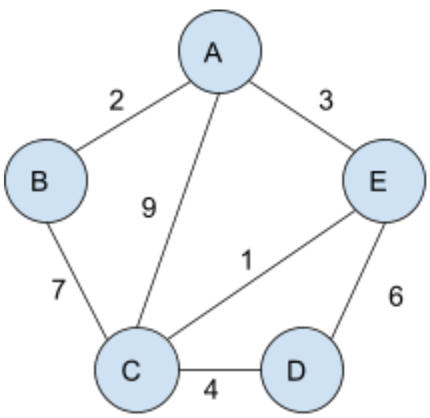
(Due Dec 7th, 2021)

No late assignments accepted

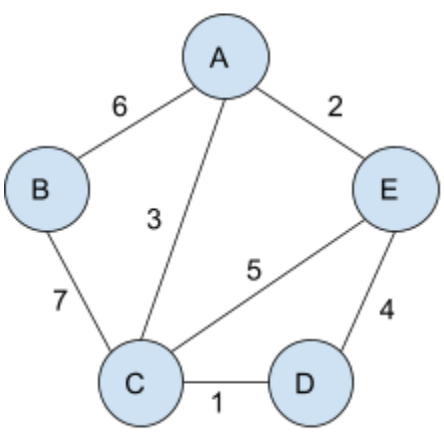
All problem/exercise numbers are for the third edition of CLRS text book

---

1. Design a greedy algorithm for making change consisting of quarters, dimes, nickels, and pennies. It will take the total number of cents as input, and output numbers of quarters, dimes, nickels, and pennies such that the total number of coins is minimum. Prove your algorithm has the greedy choice property and optimal substructure.
   Justify the running time of your algorithm.
2. How many bits are required to encode the message "abbcccxxxxyyyzz" using Huffman Codes?
3. Demonstrate Prim's algorithm for the given undirected weighted graph. (Use A as the source.)



4. If we run Kruskal's algorithm for the given graph, what will be the sequence in which edges are added to the MST?



Q1: Psedocode of my algorithm:

```
Alg:GreedyCoinChange(totalCents):
        numQuarters, numDimes, numNickels, numPennies = 0
        numQuarters = totalCents // 25
        totalCents = totalCents % 25
        numDimes = totalCents // 10
        totalCents = totalCents % 10
        numNickels = totalCents // 5
        numPennies = totalCents % 5
        return numQuarters, numDimes, numNickels, numPennie
```

Proove:

① Greedy choice property: Assume the input totalCents > numQuarters, numQuarters1 = totalCents // 25 according to my algorithm. If it is not optimal choice, the optimal choice = numQuarters2 must be smaller than numQuarters1 since (numQuarters+1)·25 > totalCents. If numQuarters2 is 1 smaller than numQuarters1, We need 2 more Dimes and 1 more nickle to make up. This is contradict with assumption. ⇒ Proved. numQuarters1 is optimal.

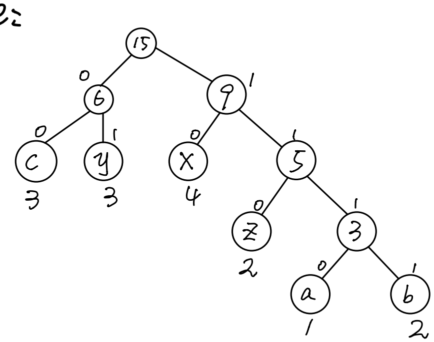
② Optimal substructure: If numQuaters1 is optimal to the problem, the subpoblem is change totalCents%25 into Dimes, knickels, and cents. The optimization problem of the same form as the original problem.

Running time = O(1)

Q2. The frequency of each letter: a=1, b=2, c=3, x=4, y=3, z=2

Hence, the Huffman's tree will be like:

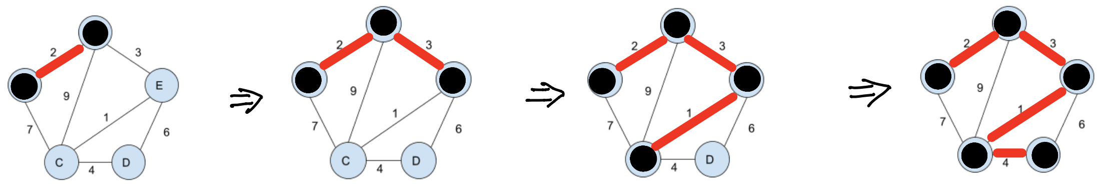
c = 00	2 bits
y = 01	2 bits
x = 10	2 bits
z = 110	3 bits
a = 1110	4 bits
b = 1111	4 bits



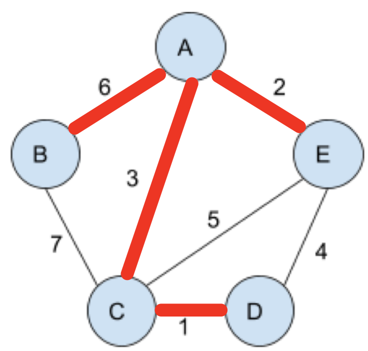
The total bits = a.freq × a.length + b.freq × b.length + c.freq × c.length + x.freq · x.length + y.freq × y.length + z.freq × z.length

$= 4\times1 + 4\times2 + 3\times3 + 2\times4 + 2\times3 + 2\times2 = 38$

Q3.



Q4.



Step1: $Find(C) \neq Find(D)$ $A = \{(C, D)\}$

Step2: $Find(A) \neq Find(E)$ $A = \{(C, D), (A, E)\}$

Step3: $Find(A) \neq Find(C)$ $A = \{(C, D), (A, E), (A, C)\}$

Step4: $Find(D) = Find(E)$ $A = \{(C, D), (A, E), (A, C)\}$

Step5: $Find(C) = Find(E)$ $A = \{(C, D), (A, E), (A, C)\}$

Step6: $Find(A) \neq Find(B)$ $A = \{(C, D), (A, E), (A, C), (A, B)\}$

Step7: $Find(B) = Find(C)$ $A = \{(C, D), (A, E), (A, C), (A, B)\}$