Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 5B (60 points) **Summer 2020, CU-Boulder**

Advice 1: For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.

Advice 2: Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

Instructions for submitting your solution:

- The solutions **should be typed**, we cannot accept hand-written solutions. Here's a short intro to **Latex**.
- In this homework we denote the asymptomatic *Big-O* notation by $\mathcal{O}$ and *Small-O* notation is represented as o.
- We recommend using online Latex editor **Overleaf**. Download the **.tex** file from Canvas and upload it on overleaf to edit.
- You should submit your work through **Gradescope** only.
- If you don't have an account on it, sign up for one using your CU email. You should have gotten an email to sign up. If your name based CU email doesn't work, try the identikey@colorado.edu version.
- Gradescope will only accept **.pdf** files (except for code files that should be submitted separately on Canvas if a problem set has them) and **try to fit your work in the box provided**.
- You cannot submit a pdf which has less pages than what we provided you as Gradescope won't allow it.

Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 5B (60 points) **Summer 2020, CU-Boulder**

Piazza threads for hints and further discussion

Piazza Threads
Question 1 Question 2 Question 3 Question 4

Recommended reading:
Graph Algorithms Intro: Ch. 22 → 22.1, 22.2, 22.3
Graph Algorithms SSSPs: Ch. 24 → 24.3

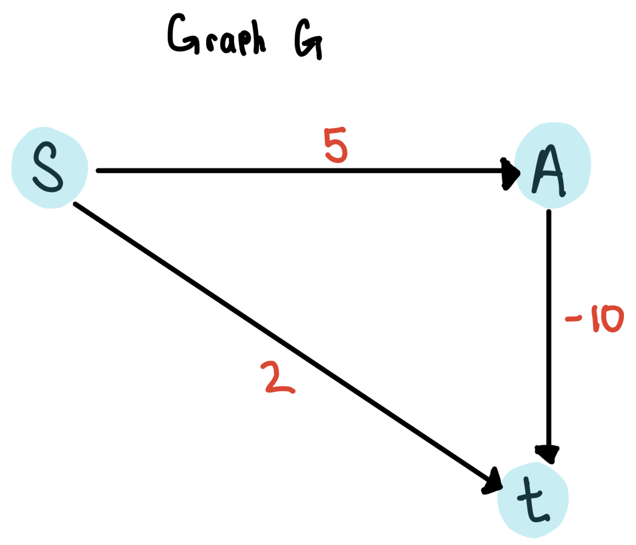
Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 5B (60 points) **Summer 2020, CU-Boulder**

1. (7.5 pts) Give an example of a simple directed, weighted graph G and identify two vertices s and t such that Dijkstra's algorithm started at s does not find the shortest $s \to t$ path.



Dijktra's algorithm does not find the shortest path when there is a negative edge. Here dijktra's algorithm would identify the shortest path from $s \to t$ as [s,t]. We can see from the simple graph G that the shortest path $s \to t$ is -5, [s,A,t].

Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 5B (60 points) **Summer 2020, CU-Boulder**

2. (7.5 pts) Suppose that you have calculated the shortest paths to all vertices from a fixed vertex $s \in V$ of an undirected graph $G = (V, E)$ with positive edge weights. If you increase each edge weight by 5, will the shortest paths from s change? Prove that it cannot change or give a counterexample if it changes.

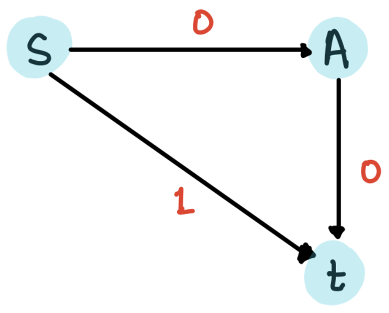
If we increase each edge by 5, the shortest path from s will change.
Suppose we have graph G with the shortest path:

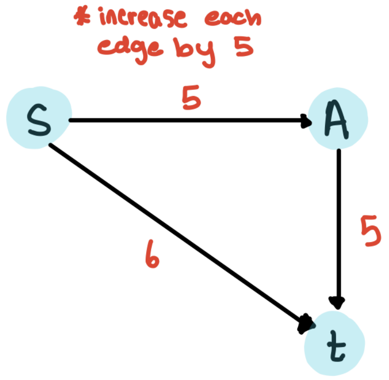
path : weight

$s \to a$: 0

$s \to t$: 1

$a \to t$: 0





Shortest Path s→t is [s, A, t]

Shortest path s→t [S, t]

We can see that the shortest path from s changes after increasing each edge weight by 5.

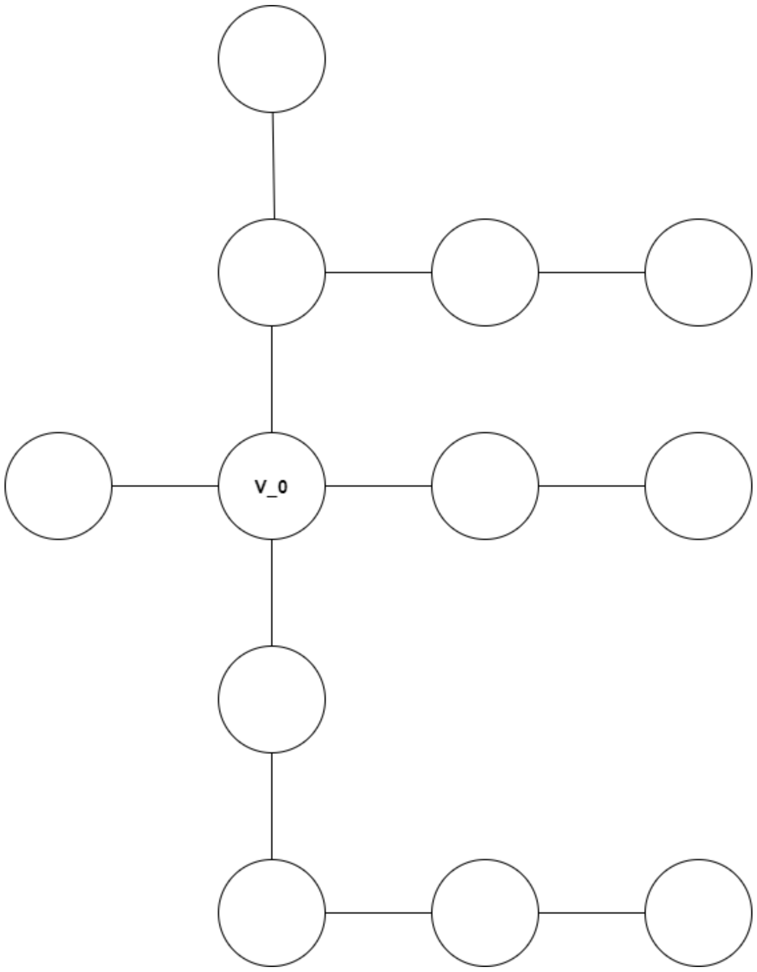
Name:	Sasha Farhat
ID:	105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 5B (60 points) **Summer 2020, CU-Boulder**

3. (20 pts) You are given an undirected tree with and a source vertex v_0. You have to the count of the number of vertices in the tree (excluding the source vertex v_0) which are at a distance less than or equal to k from the source vertex v_0.

   The inputs to your algorithm are an undirected tree in the form of an adjacency list or adjacency matrix and a source vertex v_0.
   The output should be the count of the vertices that are at a distance less than or equal to k from the vertex v_0.

   For example consider the following tree with a vertex v_0 and $k = 2$.



The output for the above tree should be the value **8**, as there are 4 neighbouring nodes at unit distance from v_0 and 4 nodes at distance of two units from v_0.

(a) (5 pts) Provide a 3-4 sentence description of how your algorithm works, including how you would maintain the distance from the source vertex v_0.

We will use a modification of BFS to traverse through every node. The algorithm will recursively call BFS for every node that has not been visited. We will compare all the nodes to the starting vertex, and analyze the distance and increment count if and only if the vertex distance is in range of k. Creating a boolean array, we will store the visited array in, which we will queue and dequeue. Return count when all distances have been checked.

Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 5B (60 points) **Summer 2020, CU-Boulder**

(b) (15 pts) Provide a well commented pseudo-code or actual code to solve the above problem.

```
class Graph:

        def __init__(self):

                # we created graph from list and must store
                #it in python dictionary
                self.graph = defaultdict(list)

        # function to add an edge to graph
        def addEdge(self,u,v):
                self.graph[u].append(v)

        # Function to print the count vertex k distance away from given initial given vertex
        def kdistBFS(self, s,k):

                # create boolean array (visited) and
                # initially mark all the vertices as false or unvisted
                visited = [False] * (len(self.graph))

                # this will be our queue for bfs
                queue = []

                #initialize the count to one because
                #we count the initial node given

                count = 1
                # Mark the given node as true
                # or visited and enqueue it
                queue.append(s)
                visited[s] = True

                while queue:

                        # Dequeue vertex from queue
                        s = queue.pop(0)

                        # enque through all the vertices within the neighboring visited node s
                        # using our boolean array we uda
                        for i in self.graph[s]:
                                if visited[i] == False:
                                        #only count the vertices that are in range of k
                                        queue.append(i)
                                        visited[i] = True
                                        if i in range(k):
                                            count=count +1
                print(count)


g = Graph()
k= 3
s = 4

#built graph from example

g.addEdge(1,2)
g.addEdge(2,1)
g.addEdge(2,6)
g.addEdge(6,2)
g.addEdge(6,7)
g.addEdge(7,6)
g.addEdge(2,3)
g.addEdge(3,2)
g.addEdge(8,3)
g.addEdge(3,8)
g.addEdge(3,9)
g.addEdge(9,3)
g.addEdge(9,10)
g.addEdge(10,9)
g.addEdge(3,4)
g.addEdge(4,3)
g.addEdge(4,5)
g.addEdge(5,4)
g.addEdge(5,11)
g.addEdge(11,5)
g.addEdge(11,12)
g.addEdge(12,11)

print ("Number of vertices within K distance away from v_0 (cell 4)")

# call kdistBFS and prints the counted vertices
g.kdistBFS(s,k)
```

Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 5B (60 points) **Summer 2020, CU-Boulder**

4. (25 pts) You have two batteries b_1, b_2, with capacities c_1 mAh, c_2 mAh respectively. Both the batteries initially have no charge. You have a charging station with infinite supply of electricity and also an earthing device that can be used to remove charge from the battery. You have a battery transfer device which has a source battery position and a target battery position. When you place two batteries in the transfer device, it instantaneously transfers as many mAh from the source battery to the target battery as possible. Thus, this device stops the transfer either when the source battery has no mAh remaining or when the destination battery is fully charged (whichever comes first). The above device can also to fully charge/discharge a battery using the charging station/earthing device.

   The goal in this problem is to determine whether there exists a sequence of transfers amongst the batteries such that in the end, a charge amount of k remains in one of the batteries, where $k \leq c_1$ and $k \leq c_2$.

   For example, consider the case where $c_1 = 4$, $c_2 = 3$ and $k = 2$.
   Let's represent the charge in both the batteries in the form of a tuple (x, y), where x is the charge of b_1 and y is the charge of b_2. The following sequence of transfers are possible.

   - (0, 0) initially both the batteries have no charge.
   - (4, 0) charged b_1 using the charging station.
   - (1, 3) transferred the charge from b_1 to b_2.
   - (1, 0) discharged b_2 using earthing device.
   - (0, 1) transferred the charge from b_1 to b_2.
   - (4, 1) charged b_1 using the charging station.
   - (2, 3) transferred the charge from b_1 to b_2.

Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 5B (60 points) **Summer 2020, CU-Boulder**

(a) (5 pts) Rephrase this is as a graph problem. Give a precise definition of how to model this problem as a graph, including how to define a vertex and identify it's neighbouring vertices, and state the specific question about this graph that must be answered.

The given problem will be represented as a tuple (x, y) where x represents the amount of charge in the 4-capacitor battery and y represents the amount of charge in the 3-capacitor battery.In this graph representation, any node of the form (*,k) or (k,*) represents a solution to the problem of k charge, since one of the batteries holds exactly k gallons. So, given two batteries that hold b_1 and b_2 charge, and an initial configuration (x,y), we find the sequence of operations that has the least weight and produces k charge in one of the batteries.

(b) (5 pts) Provide a 3-4 sentence description of how your algorithm works

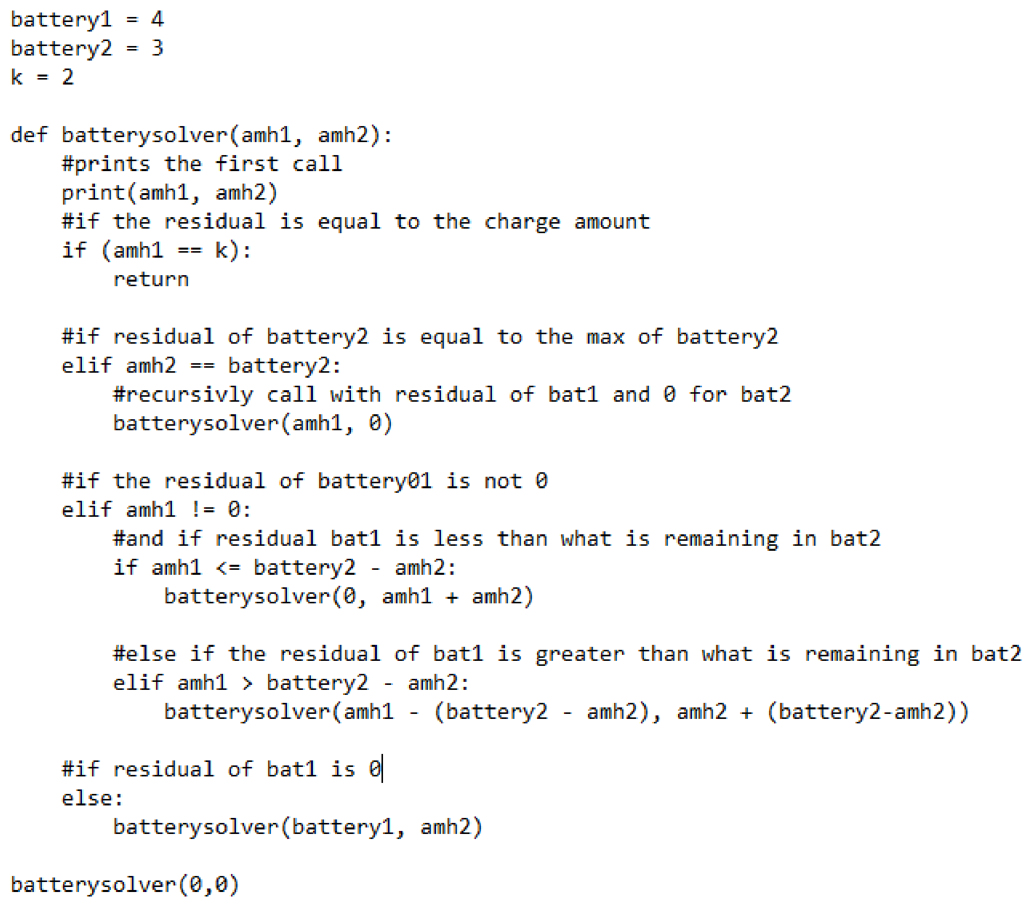
My algorithm will perform one of the 5 operations:
(1) It will empty the charge in from battery1
(2) It will empty the charge in battery2
(3) It will fill battery1 to the max capacity
(4)Give charge from battery1 to battery2 till battery 2 is full pr battery1 has no charge left to give
(5) Give charge from battery2 to battery1 till battery 1 is full or battery2 has no charge left to give
The algorithm will start with an initial state of (0,0), then will follow the sequence of transfers based on the weight of operations.

Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**
Homework 5B (60 points) **Summer 2020, CU-Boulder**

(c) (15 pts) Provide a well commented pseudo-code or actual code to solve the above problem.



```
battery1 = 4
battery2 = 3
k = 2

def batterysolver(amh1, amh2):
    #prints the first call
    print(amh1, amh2)
    #if the residual is equal to the charge amount
    if (amh1 == k):
        return

    #if residual of battery2 is equal to the max of battery2
    elif amh2 == battery2:
        #recursivly call with residual of bat1 and 0 for bat2
        batterysolver(amh1, 0)

    #if the residual of battery01 is not 0
    elif amh1 != 0:
        #and if residual bat1 is less than what is remaining in bat2
        if amh1 <= battery2 - amh2:
            batterysolver(0, amh1 + amh2)

        #else if the residual of bat1 is greater than what is remaining in bat2
        elif amh1 > battery2 - amh2:
            batterysolver(amh1 - (battery2 - amh2), amh2 + (battery2-amh2))

    #if residual of bat1 is 0
    else:
        batterysolver(battery1, amh2)

batterysolver(0,0)
```

Name: Sasha Farhat

ID: 105887541

CSCI 3104, Algorithms **Escobedo & Jahagirdar**

Homework 5B (60 points) **Summer 2020, CU-Boulder**

5. ***Extra Credit (5% of total homework grade)*** *For this extra credit question, please refer the leetcode link provided below or click here. Multiple solutions exist to this question ranging from brute force to the most optimal one. Points will be provided based on Time and Space Complexities relative to that of the most optimal solution.*

   Please provide your solution with proper comments which carries points as well.

   https://leetcode.com/problems/course-schedule-ii/

   ```
   Replace this text with your source code inside of the .tex document
   ```