CISC-235 Data Structures W21
Assignment 4

March 24, 2022

1 Problems

1.1 Random Graph Generation: 10 points

Create a graph and a node class. Your graph class should have an initialization function to create a random connected graph with a specified number of vertices, and random weights on all the edges, such that the edge-weights all come from a specified range of integers.

The construction of random graphs is a huge topic and many approaches have been developed. For the purposes of this assignment, we suggest the following method:

```
Let the set of vertices be {1, 2, ..., n}
for i = 2 .. n:
  x = randint(1,i-1) # x is a random integer in the range [1 .. i-1]
  let S be a randomly selected sample of x values from the set {1, ..., i-1}
  for each s in S:
    w = random(10,100)
    add an edge between vertex i and vertex s, with weight w
```

1.2 Breadth-First Search: 20 points

Implement the Breadth First Search algorithm, modifying it so that it randomly chooses a start vertex, and returns the total of the weights of the edges that it selects.

Hint: the key part of this modification looks something like this:

```
for each neighbour y of x:
  if y is not visited:
    mark y visited
    Q.append(y)
    total += weight(x,y)
```

Test your code using the following graph:

	node_0	node_1	node_2	node_3	node_4	node_5
node_0	0	15	0	7	10	0
node_1	15	0	9	11	0	9
node_2	0	9	0	0	12	7
node_3	7	11	0	0	8	14
node_4	10	0	12	8	0	8
node_5	0	9	7	14	8	0

Note: you decide how you want to create the above test graph, either by adding edges one by one, or ideally you can create another initialization function for your Graph class. This initialization function takes a txt file as input and initializes an empty graph. Your input file should save a matrix representation of the test graph.

1.3 Breadth-First-Search on Directed Graph: 10 points

Briefly describe how you plan to modify your previous code so that it can perform BFS on a directed graph.

1.4 Prim's MST algorithm: 20 points

Implement Prim's Minimum Spanning Tree algorithm, so that it returns the total of the weights of the edges that it selects. Test your implementation on the same graph as was specified in 1.2.

1.5 Compare BFS with Prim's algorithm: 30 points

Perform the below experiments:

```
for n = 20, 40, 60:
    repeat k times: #k is an input parameter
      generate a random graph with n vertices
      use BFS to find a spanning tree (let its total weight be B)
      use Prim to find a spanning tree (let its total weight be P)
      compute Diff #Diff is the percentage by which B is larger than P
    compute the average of the values of Diff for this value of n
```

Report the average value of Diff for each value of n.

For example, if B=20, P=10, Diff = $(20 - 10)/10 = 100\%$

1.6 Style: 10 points

You must comment your code. You should submit runnable code, i.e., providing test files so that TA can run your code.

Assignment Requirements This assignment is to be completed individually (we have the right to run any clone detection tool). You need submit your documented source code (we don't need the input folder and query file). If you have multiple files, please combine all of them into a .zip archive and submit it

to OnQ system. The .zip archive should contain your student number. Helper functions and classes are allowed as long as you write them by yourself.