Week 9 Exercises Submit

Sandra Batista

1.1–1.2

Exercise 1: BFS Analysis

1. What is the loop invariant of BFS?
2. What is the runtime of BFS?

```
BFS(G,u)
1  for each vertex v
2       pred[v] = nil, d[v] = inf.
3  Q = new Queue
4  Q.enqueue(u),  d[u]=0
5  while Q is not empty
6     v = Q.front(); Q.dequeue()
7     foreach neighbor, w, of v:
8          if pred[w] == nil // w not found
9               Q.enqueue(w)
10              pred[w] = v,  d[w] = d[v] + 1
```

Exercise 2: DFS Analysis

1. What is the loop invariant of DFS?
2. What is the runtime of DFS?

```
DFS-Visit (G, start_node)
1    for each vertex u
2      u.color = WHITE
3      u.pred = nil
4    st = new Stack
5    st.push_back(start_node)
6    while st not empty
7       u = st.top(); st.pop()
8       if u.color == WHITE
9          u.color = GRAY
10         foreach vertex v in Adj(u) do
11            st.push_back(v)
```

What is the run-time of Dijkstra's algorithm?

```
1.    SSSP(G, s)
2.      PQ = empty PQ
3.      s.dist = 0;  s.pred = NULL
4.      PQ.insert(s)
5.      For all v in vertices
6.          if v != s then v.dist = inf;
7.          PQ.insert(v)
8.      while PQ is not empty
9.          v = min(); PQ.remove_min()
10.         for u in neighbors(v)
11.             w = weight(v,u)
12.             if(v.dist + w < u.dist)
13.                 u.pred = v
14.                 u.dist = v.dist + w;
15.                 PQ.decreaseKey(u, u.dist)
```

Starter code:
https://github.com/sandraleeusc/csci104_fall2020_lecture/blob/master/directedGraph.cpp

In this exercise you will implement a directed graph using adjacency lists.

You may assume that all the nodes are labeled with numbers from 1 to n where n is the number of nodes in the graph.

You are also given a struct for a node on the adjacency list.

You must complete the implementation of the DirectedGraph in the starter code.

5. Generating Subsets

Starter code:
https://github.com/sandraleeusc/csci104_fall2020_lecture/blob/master/printsubset.cpp

Write a function, print_subsets, that is given a vector of strings, stringSet, and integer, k, outputs all subsets of stringSet of size k. You may assume that k is greater than 0 and less than or equal to the size of stringSet. Your implementation should be recursive and you may use helper functions. You may also use loops if absolutely necessary.

Suppose stringSet is the set, {”cat”, ”dog”, ”rabbit”, ”robot”} and k is 3.

Your function should print the following:

cat dog rabbit

cat dog robot

cat rabbit robot

dog rabbit robot

```
void print_subsets (vector<string> stringSet, int k);
```

6. Undirected Graphs and Adjacency Matrices

Starter code:

https://github.com/sandraleeusc/csci104_fall2020_lecture/blob/master/hgraph.cpp

In this exercise you will implement an undirected graph using an adjacency matrix.

In this problem the nodes will be labeled with strings, so you will need to figure out how to use those strings with the matrix.