Algorithms & Data Structures
SI3 - Polytech Nice-Sophia - Edition 2018

Lab #9: Minimum Spanning Tree

This lab will give you practice about undirected weighted graphs and minimum spanning tree algorithms. For all parts you are to use the provided material:

- partition.py: skeleton file for the Partition class which implements disjoint sets
- lab9.py: function skeletons and main script for testing

Part 1

In this part, you are to practice and implement disjoint sets.

Part 1.1: union by size

Starting with the forest of 1-node up-trees:

```
{0},{1},{2},{3},{4},{5},{6},{7},{8}
```

draw the resulting up-trees of the following operations:

```
union(0,1), union(2,3), union(4,5), union(0,4), union(2,6), union(7,8),
union(0,7), union(2,0)
```

where union is the union by size studied in lecture.

Part 1.2: path compression

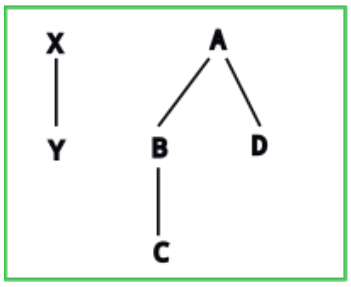
Starting with the up-trees from part 1, draw the updated up-trees after we perform the following operations:

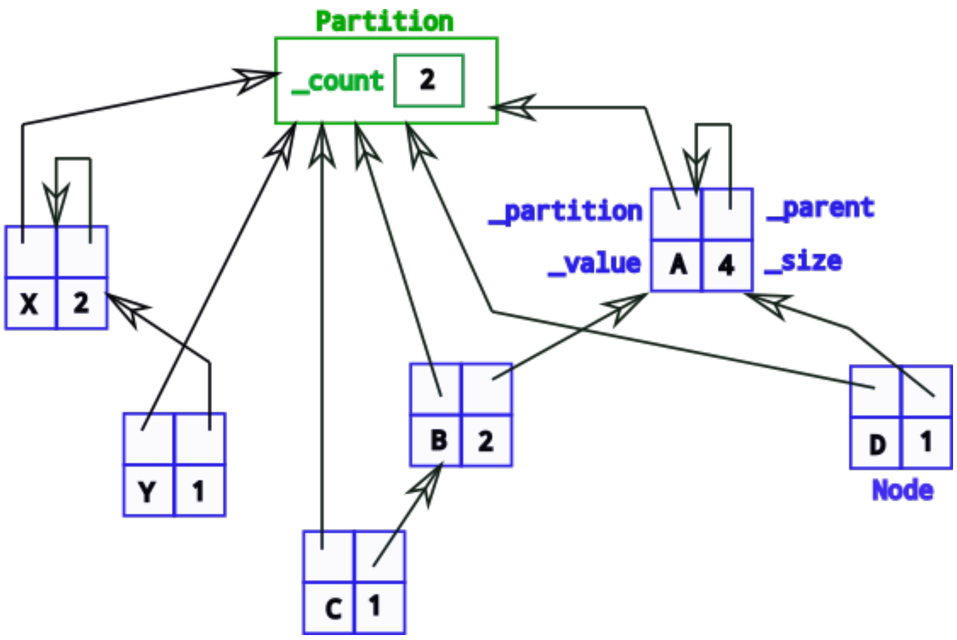
```
find(1), find(3), find(7)
```

where `find` is the find with path compression studied in class.

Part 1.3: disjoint sets implementation

Complete the file `partition.py` which defines the class `Partition` to implement disjoint sets. The class `Partition` and its internal class `Node` relates in the following way:



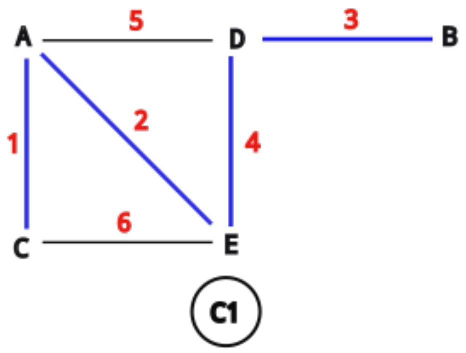


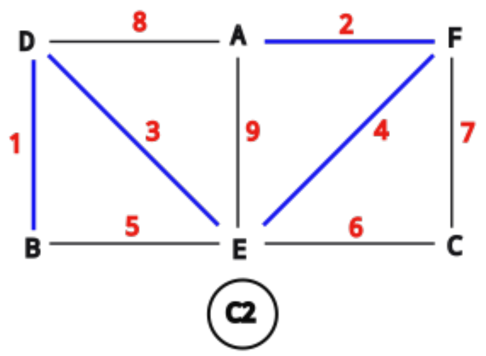
Part 2

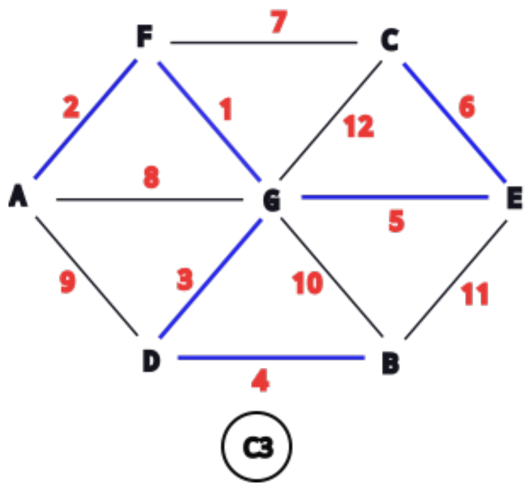
In this part, you are to implement both the Prim and the Kruskal algorithms. For this, you are to complete the two function skeletons:

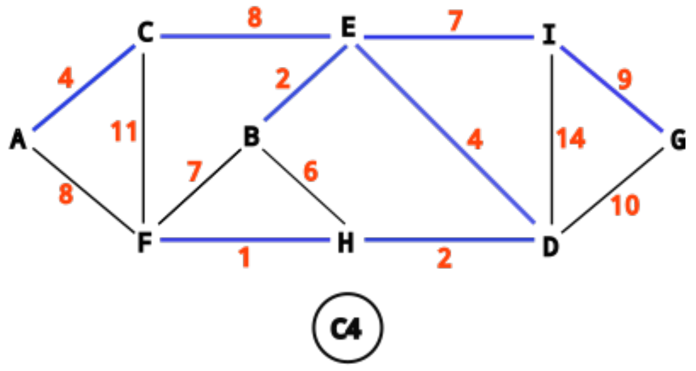
- the `prim`: function: this file implements the Prim algorithm
- the `kruskal`: function: this file implements the Kruskal algorithm

You can test your functions with various graphs including the one reviewed in class:





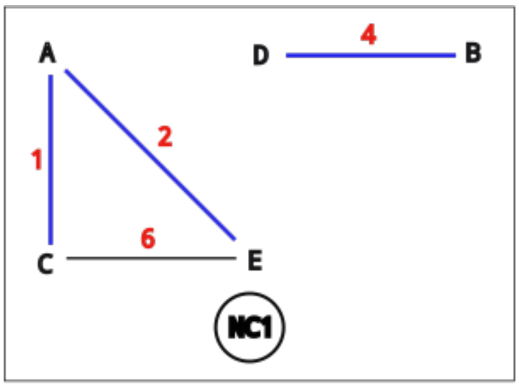


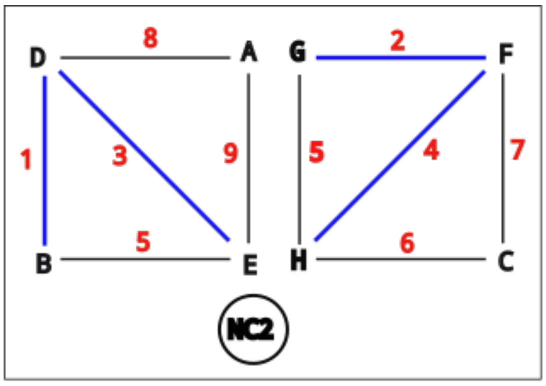


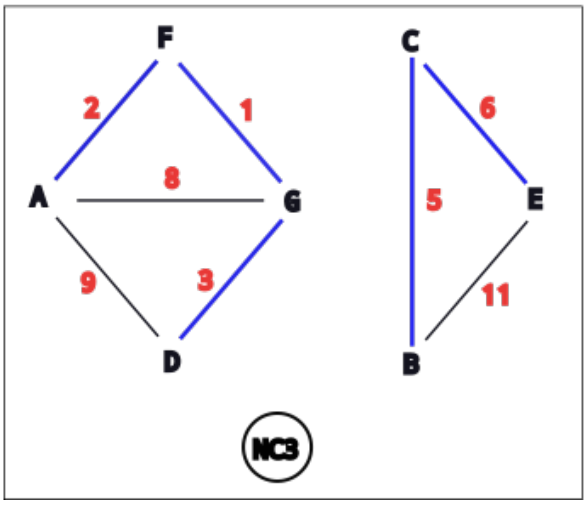
Part 3

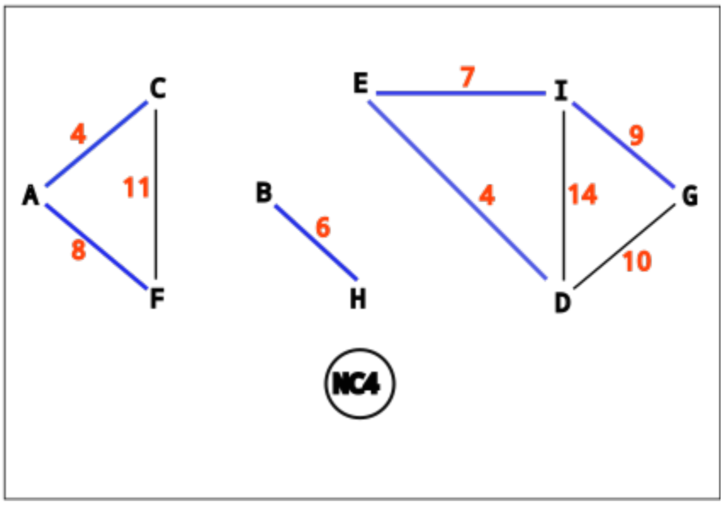
In this part, we consider the following problem: how can we change the Prim and Kruskal algortihms so that they can work on weighted undirected graphs which are not connected?

We can notice that when the graph is not connected, we don't compute a (minimum spanning) *tree* anymore but a (minimum spanning) *forest* made of several trees. Clearly the number of trees is equal to the number of connected components. If there is only one connected component (i.e. the graph is connected) then the algorithm is the regular one (either Prim or Kruskal) and the result is a minimum spanning tree. For example, the following graph has 3 connected components and then a minimum spanning forest of 3 minimum spanning trees (the blue edges):









You are to implement a new version of both Prim and Kruskal algorithms to work on weighted undirected graphs which are not connected. You have to complete the two functions:

- the `primForest` functon: this function implements the Prim algorithm adapted to our new problem. We call it `primForest` because the result is no longer a minimum spanning tree but a minimum spanning forest.
- the `kruskalForest` function: this function implements the Kruskal algorithm adapted to our new problem

Your functions should work with connected and non connected weighted undirected graphs.