# **Self-Practice Week 1 - Fundamentals (part 1)**

# **Developing Algorithms**

The goal of this assignment is to practice designing, implementing and testing basic algorithms.

## **Exercise 1 – Union-Find**

Implement the Union-Find algorithm seen at lectures in Python. Test its functioning.

## **Exercise 2 – Social Network Connectivity**

Given a social network containing N members and a log file containing a sequence of friendships requests, design an algorithm to determine the earliest time at which all members are connected (i.e., every member is a friend of a friend of a friend ... of a friend). Assume that the log file is temporally sorted, and that friendship is an equivalence relation. What is the running time of your algorithm?

## *Hint*

Use Union-Find to model friendships

## **Exercise 3 – Extended Union-Find**

Extend Union-Find with a method find so that find(i) returns the largest element in the connected component containing i. For example, if one of the connected components is {1,2,6,9}, then the find method should return 9 for each of the four elements in the connected components. The operations union(),  connected(), and find() should all take logarithmic time or better.

## *Hint*

Maintain an extra array that stores, for each i, the largest element in the connected component containing i.