FIT3155: Week 6 tutorial
Covering concepts from Weeks 4-5

Objectives: The tutorials, in general, give practice in problem solving, in analysis of algorithms and data-structures, and in logic useful in the above.

Instructions to the class: Prepare your answers to the questions **before** the tutorial. It will probably not be possible to cover all questions unless the class has prepared them all in advance.

Instructions to Tutors:

i. The purpose of the tutorials is not to solve the practical exercises.

ii. The purpose is to check answers, and to discuss particular sticking points, not to simply make answers available.

1. Consider a disjoint set data structure involving 17 elements labeled $\{0, \ldots, 16\}$. Consider the following sequence of operations:

   - union(1,2)
   - union(3,4)
   - union(3,5)
   - union(1,7)
   - union(3,6)
   - union(8,9)
   - union(1,8)
   - union(3,10)
   - union(3,11)
   - union(3,12)
   - union(3,13)
   - union(14,15)
   - union(16,0)
   - union(14,16)
   - union(1,3)

- union(1,14)

Work out on paper the visual tree representation and its corresponding parent array representation after each union operation when the union is implemented as:

(a) union **by size** without path compression.

(b) union **by height** without path compression.

(c) union by height **with** path compression.

2. Show that for a disjoint set data structure of N elements using union **by height**, the depth of the underlying tree is bounded by $O(\log N)$.

3. Design a new function `remove`(x) that is added to the disjoint set data structure which allows x to be removed from a set and be placed independently as it own.

4. Design a disjoint set data structure that implements *partial* path compression during any `find`(x) operation, where every alternate node on the path from x to the leader/root node points to its grandparent.

5. Draw a binomial heap containing 29 elements, using the tree representation.

6. Draw the binomial heap that results when the following elements/keys are inserted (starting from an initially empty heap): $\{F, B, T, Z, R, V, F, T, U, J, P, O\}$.

7. Clarify any conceptual difficulties you may still have about Ukkonen's algorithm.

```
-=o0o=-
  END
-=o0o=-
```