ADS 2022 spring Week 1 Exercises

Exercises for Algorithms and Data Structures at ITU. The exercises are from *Algorithms, 4th Edition* by Robert Sedgewick and Kevin Wayne unless otherwise specified. Color-coding of difficulty level and alterations to the exercises (if any) are made by the teachers of the ADS course at ITU.

1.1.14 - Green Design an algorithm that takes an integer value N as argument and returns the largest integer not larger than the base-2 logarithm of N. Do not use a math library.

1.5.1 - Green Show the contents of the id[] array and the number of times the array is accessed for each input pair when you use quick-find for the sequence 9-0 3-4 5-8 7-2 2-1 5-7 0-3 4-2.

1.5.2 - Green Do Exercise 1.5.1, but use quick-union (page 224). In addition, draw the forest of trees represented by the id[] array after each input pair is processed.

1.5.3 - Green Do Exercise 1.5.1, but use weighted quick-union (page 228).

1.5.8 - Yellow Give a counterexample that shows why this intuitive implementation of union() for quick-find is not correct:

```python
# Python

def union (self, p: int, q: int) -> None:
  if self.connected(p, q):
    return

  # Rename p's component to q's name.
  for i in range(0, len(id)):
    if id[i] == id[p]:
      id[i] = id[q]
  self._count -= 1
```

```java
// Java

public void union(int p, int q) {
    if (connected(p, q)) return;

    // Rename p's component to q's name.
    for (int i = 0; i < id.length; i++)
        if (id[i] == id[p]) id[i] = id[q];
    count--;
}
```

1.5.9 - Red Draw the tree corresponding to the id[] array depicted below. Can this be the result of running weighted quick-union? Explain why this is impossible or give a sequence of operations that results in this array.

```
i      0 1 2 3 4 5 6 7 8 9
id[i] 1 1 3 1 5 6 1 3 4 5
```