**INFO 6205**

**Program Structures & Algorithms**

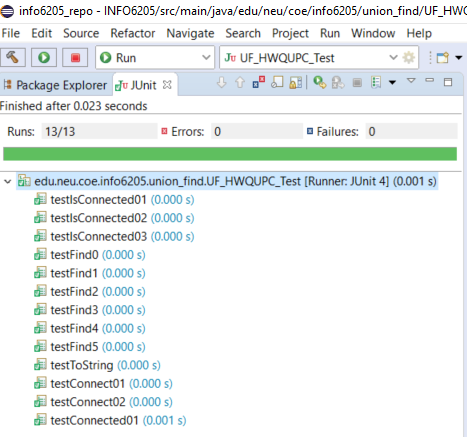
**Fall 2020**

**Assignment 3**

**Git repository:** <https://github.com/songyin79/INFO6205/tree/master/src/main/java/edu/neu/coe/info6205/union_find>

* **Step 1:**

1. **Implement height-weighted Quick Union with Path Compression** <https://github.com/songyin79/INFO6205/tree/master/src/main/java/edu/neu/coe/info6205/union_find>
2. **Test cases in UF\_HWQUPC\_Test.java have passed**



* **Step 2:**

**Using your implementation of UF\_HWQUPC, develop a UF ("union-find") client that takes an integer value n from the command line to determine the number of "sites." Then generates random pairs of integers between 0 and n-1, calling connected() to determine if they are connected and union() if not. Loop until all sites are connected then print the number of connections generated. Package your program as a static method count() that takes n as the argument and returns the number of connections; and a main() that takes n from the command line, calls count() and prints the returned value. If you prefer, you can create a main program that doesn't require any in put and runs the experiment for a fixed set of n values. Show evidence of your run(s).**

UF\_Driver.java is implemented to measure the relationship between n and m. it runs 00 different values of n, from n = 100 to n = 30000. According to the output of n and generated pairs m: <https://github.com/songyin79/INFO6205/blob/master/assignment3_output.csv>, the average value of the coefficient is 0.58.

* **Step 3:**

**Determine the relationship between the number of objects (n) and the number of pairs (m) generated to accomplish this (i.e. to reduce the number of components from n to 1). Justify your conclusion.**

m = 0.58 \* n \* log(n).

According to the output of n and generated pairs m: <https://github.com/songyin79/INFO6205/blob/master/assignment3_output.csv>, the average value of the coefficient is 0.58.