Program Structures and Algorithms
Spring 2024

NAME: Aneesh Arunjunai Saravanan
NUID: 002675639
GITHUB LINK: https://github.com/aneesharunjunai/INFO6205

Task: Assignment 4 – WQUPC

Step 1:
(a) Implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF_HWQUPC. All you have to do is to fill in the sections marked with // TO BE IMPLEMENTED ... // ...END IMPLEMENTATION.

(b) Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).

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 input and runs the experiment for a fixed set of n values. Show evidence of your run(s).

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 in terms of your observations and what you think might be going on.

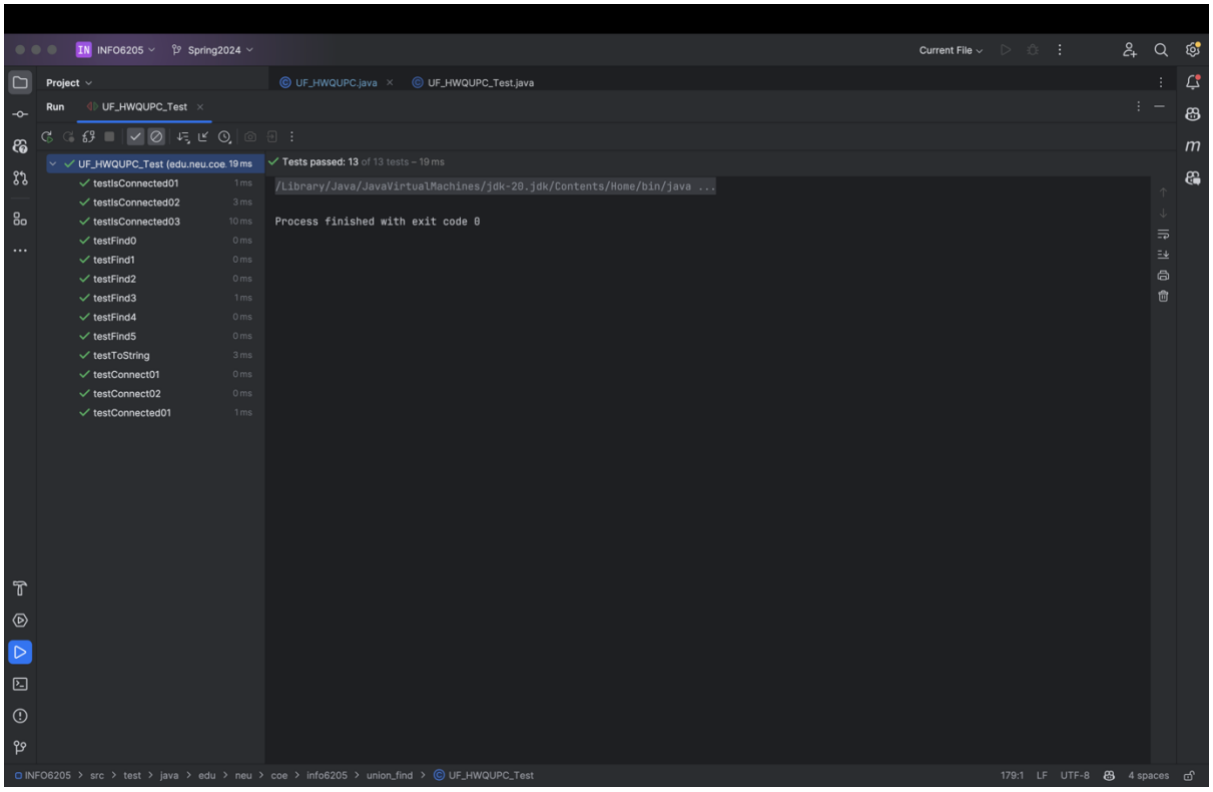
NOTE: although I'm not going to tell you in advance what the relationship is, I can assure you that it is a simple relationship.

Don't forget to follow the submission guidelines. And to use sufficient (and sufficiently large) different values of n.

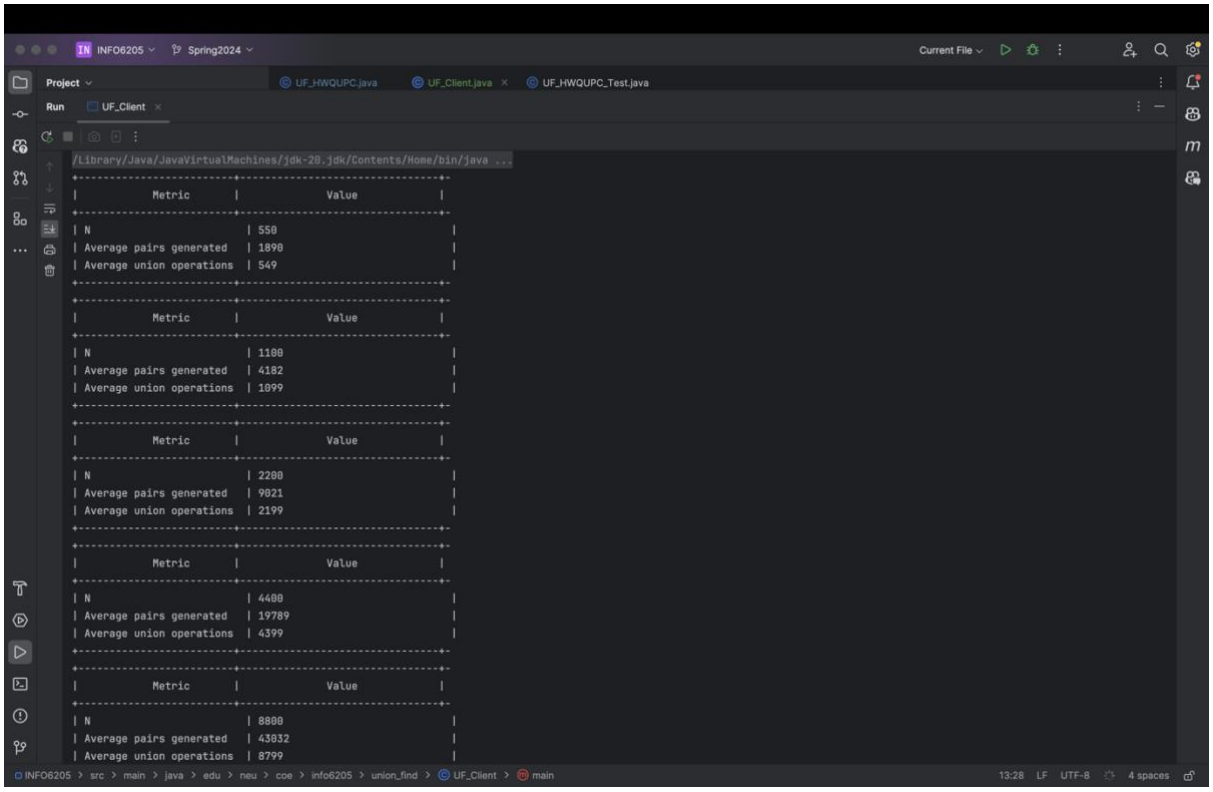
Observation & Conclusion:

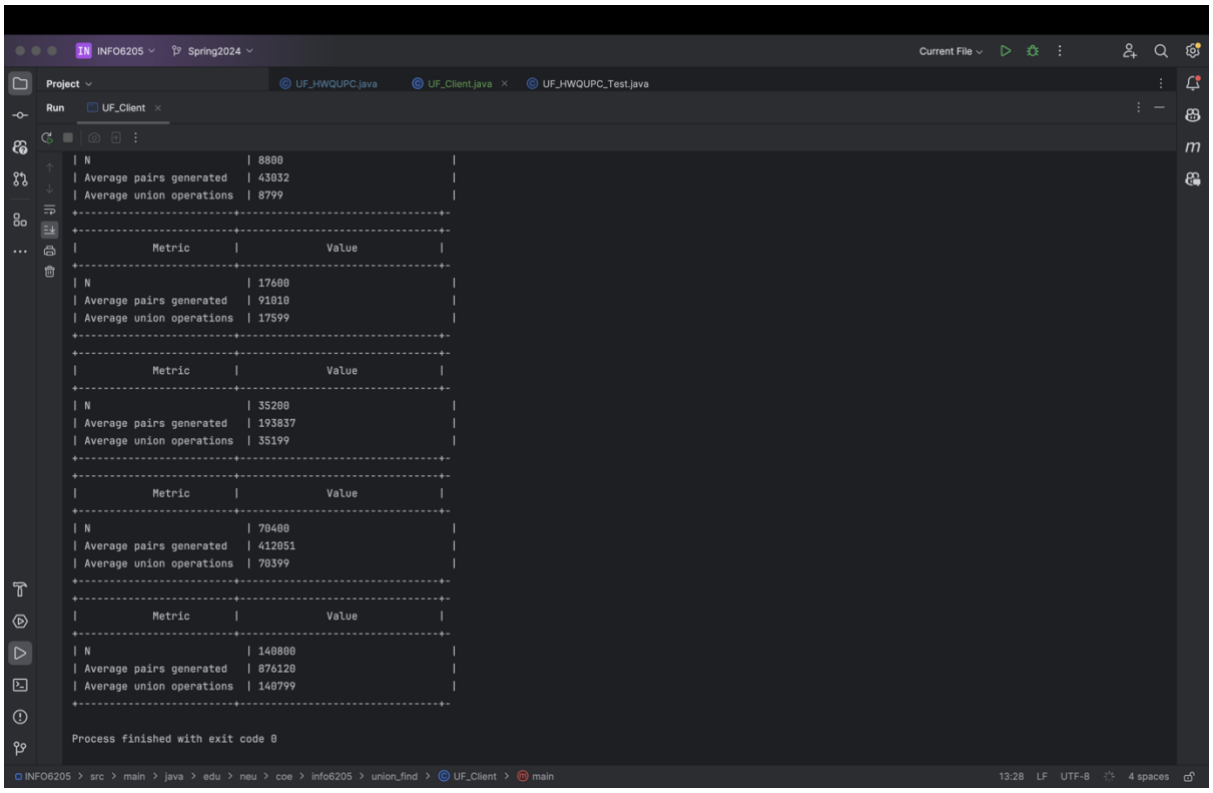
The experiments conducted demonstrate a clear linear relationship between the number of objects (n) and the number of pairs (m) generated to reduce the number of components from n to 1. This observation suggests that the Height-Weighted Quick Union with Path Compression (HWQUPC) algorithm exhibits linear behavior. As the number of objects increases, the number of pairs required to connect them into a single component increases linearly. This linear relationship aligns with the efficiency of HWQUPC, where the number of pairs generated scales linearly with the number of objects, making it a linear-time algorithm for union-find operations.

Unit Test Screenshot(s):



Console Output:





Benchmark Screenshot:

Metric	Value
N	550
Average pairs generated	1890
Average union operations	549
N	1100
Average pairs generated	4182
Average union operations	1099
N	2200
Average pairs generated	9021
Average union operations	2199
N	4400
Average pairs generated	19789
Average union operations	4399
N	8800
Average pairs generated	43032
Average union operations	8799
N	17600
Average pairs generated	91010
Average union operations	17599
N	35200
Average pairs generated	193837
Average union operations	35199
N	70400
Average pairs generated	412051
Average union operations	70399
N	140800
Average pairs generated	876120
Average union operations	14079