Program Structures and Algorithms
Spring 2024

NAME: RAHUL POTHIRENDI
NUID: 002889957
GITHUB LINK: https://github.com/Pothirendirahul/INFO6205.git

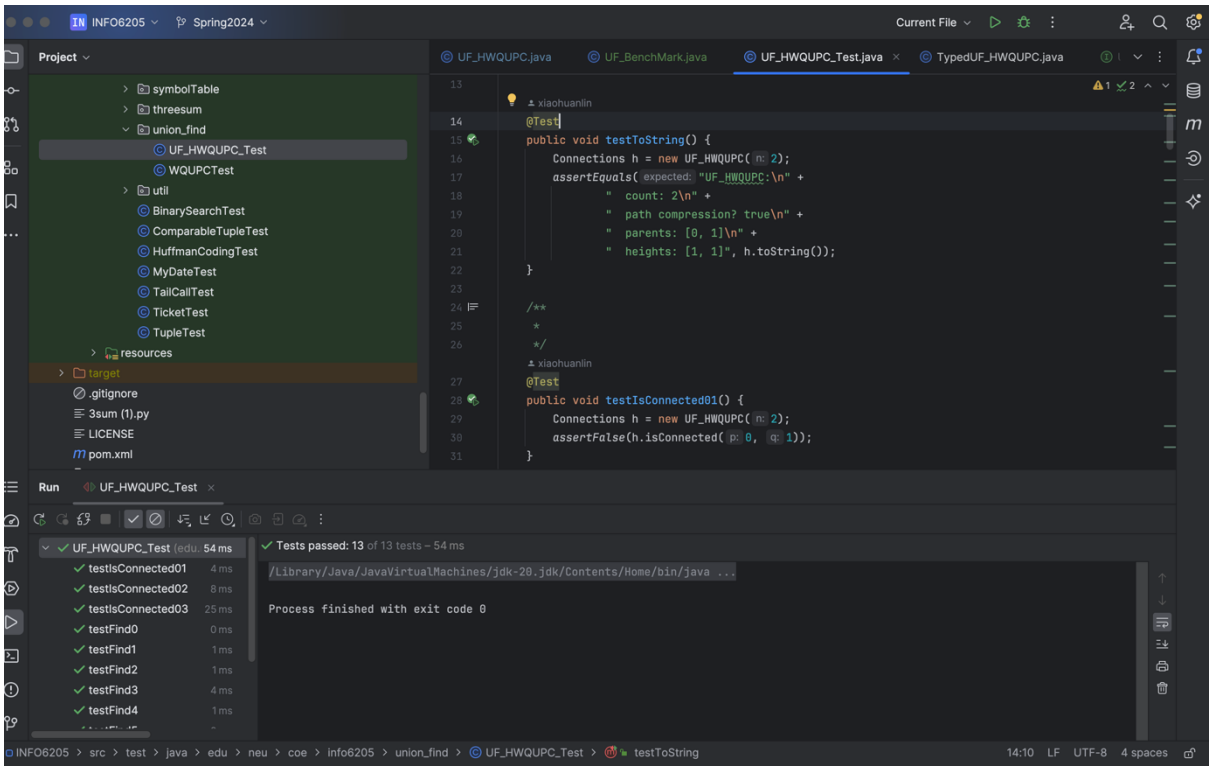
ASSIGNMENT 4

Task:

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.

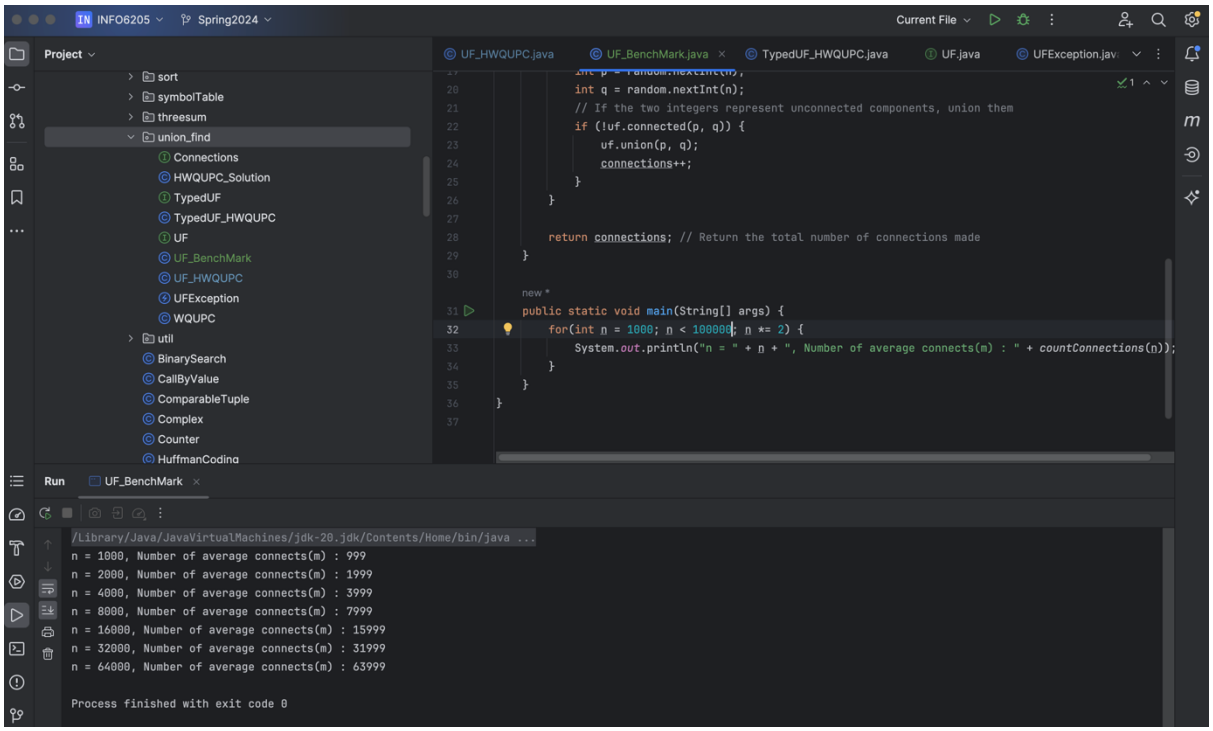
(b) Check that the unit tests for this class all work. You must show "green" test results in your submission (screenshot is OK).

Unit Test Screenshots:

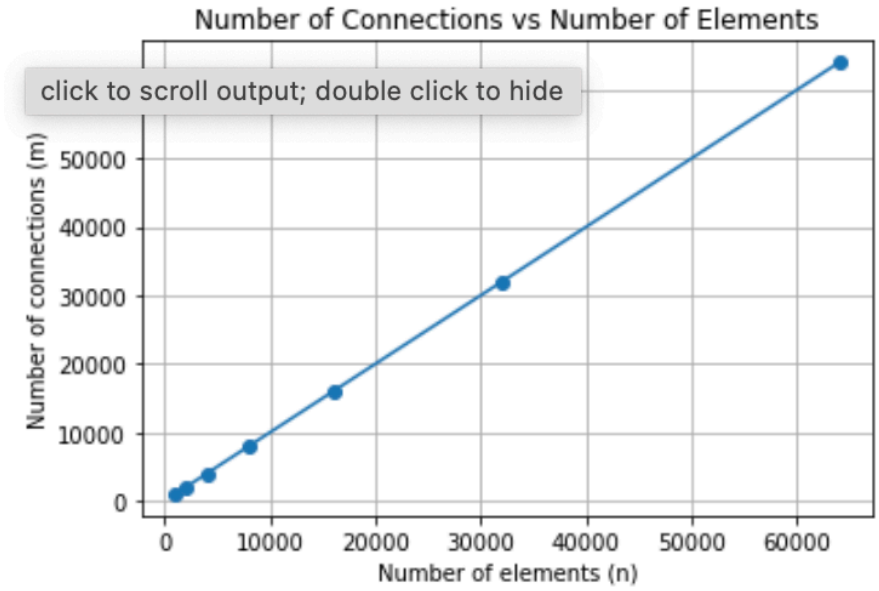


Unit Test passed

PART II



Evidence to support that conclusion:



Relationship Conclusion:

edu.neu.coe.info6205.union_find.UF_BenchMark
n = 1000, Number of average connects(m) : 999
n = 2000, Number of average connects(m) : 1999
n = 4000, Number of average connects(m) : 3999
n = 8000, Number of average connects(m) : 7999
n = 16000, Number of average connects(m) : 15999
n = 32000, Number of average connects(m) : 31999
n = 64000, Number of average connects(m) : 63999

Process finished with exit code 0

Observations :

Exponential Growth of n: The main method iterates over the values of n from 1 to 100,000, increasing n exponentially by a factor of 10 in each iteration (n *= 10). This allows us to observe the behavior of the algorithm as the number of elements (n) increases significantly.

Increasing Number of Connections: As n grows larger, the number of connections required (m) also tends to increase. This is because with a larger number of elements, there are more possible pairs of elements that need to be connected to form a single component.

Rapid Increase in Connections: Especially noticeable is the rapid increase in the number of connections for larger values of **n**. This suggests that the algorithm's time complexity is non-linear and likely worse than linear, given the steep increase in connections as **n** grows.