Feifan Tai(NUID 002138099)

**Your task is**

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.

**Relationship Conclusion：**

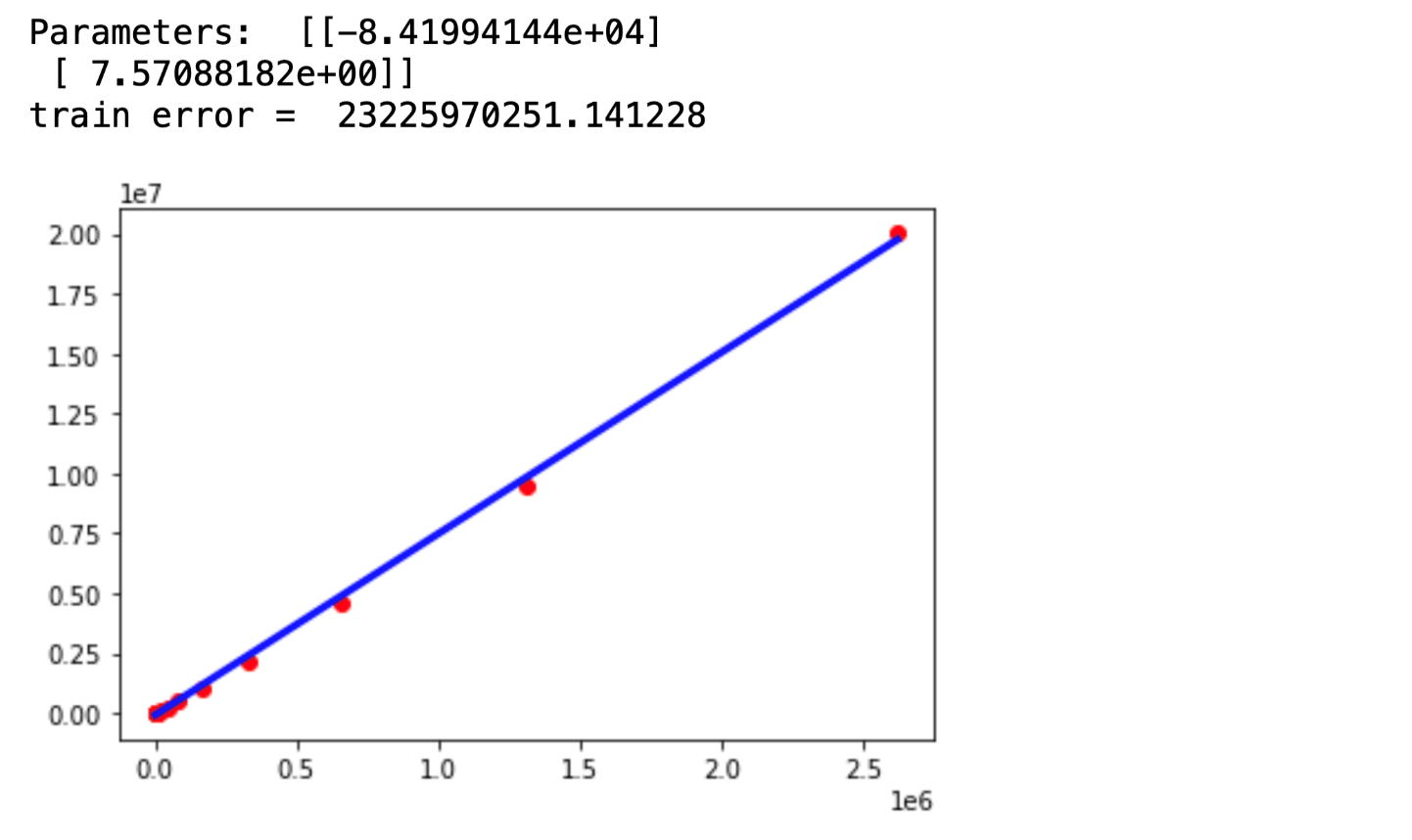
**The relationship between N and M is close to linear.**

M=a\*N+b

In my test b is close to-8.41994e+04, and b is close to a 7.57

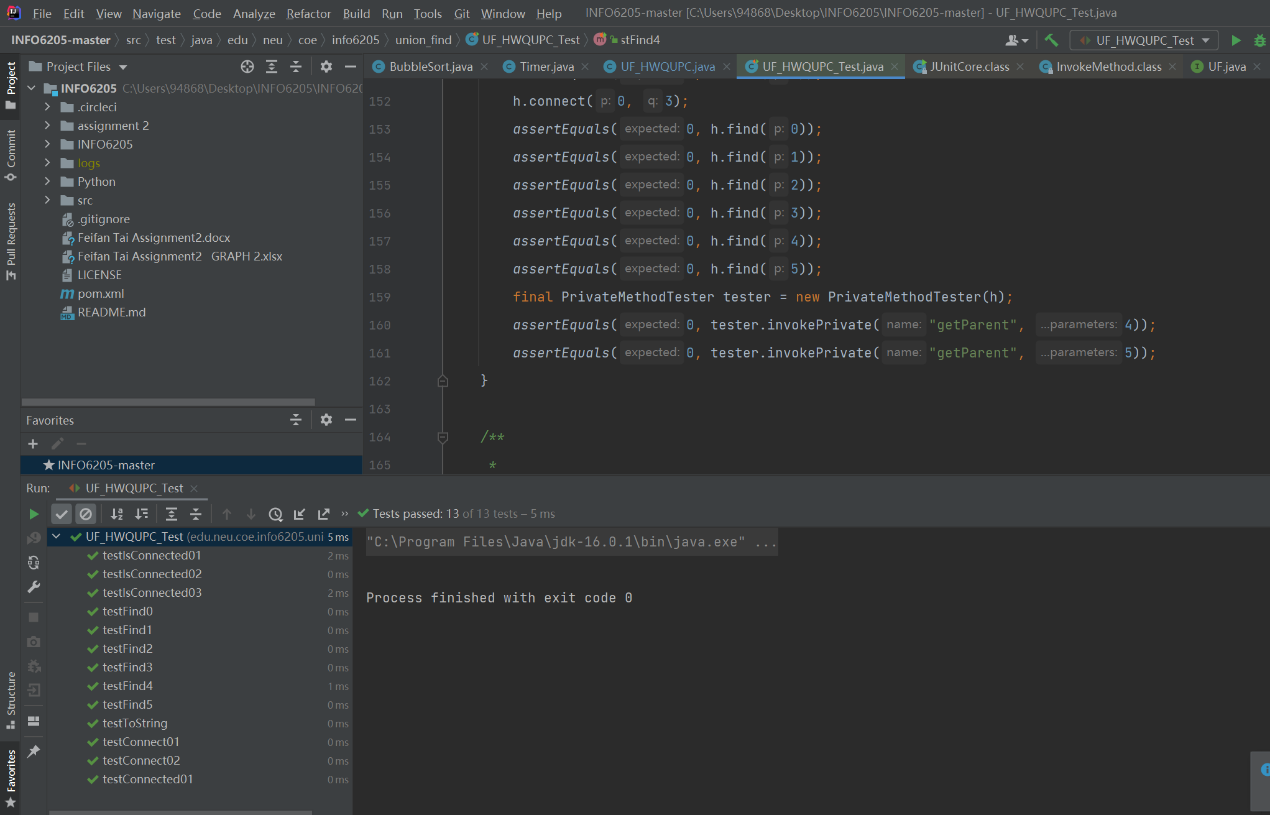
**Evidence to support the conclusion:**

**This is a graph fitted by linear regression. This experiment used doubling method to run N. Every N will run 100 times and get the average of M. Due to the large interval of selected data, the error is very large. But the fit is good that I think the relationship between m and n can be approximated as linear**

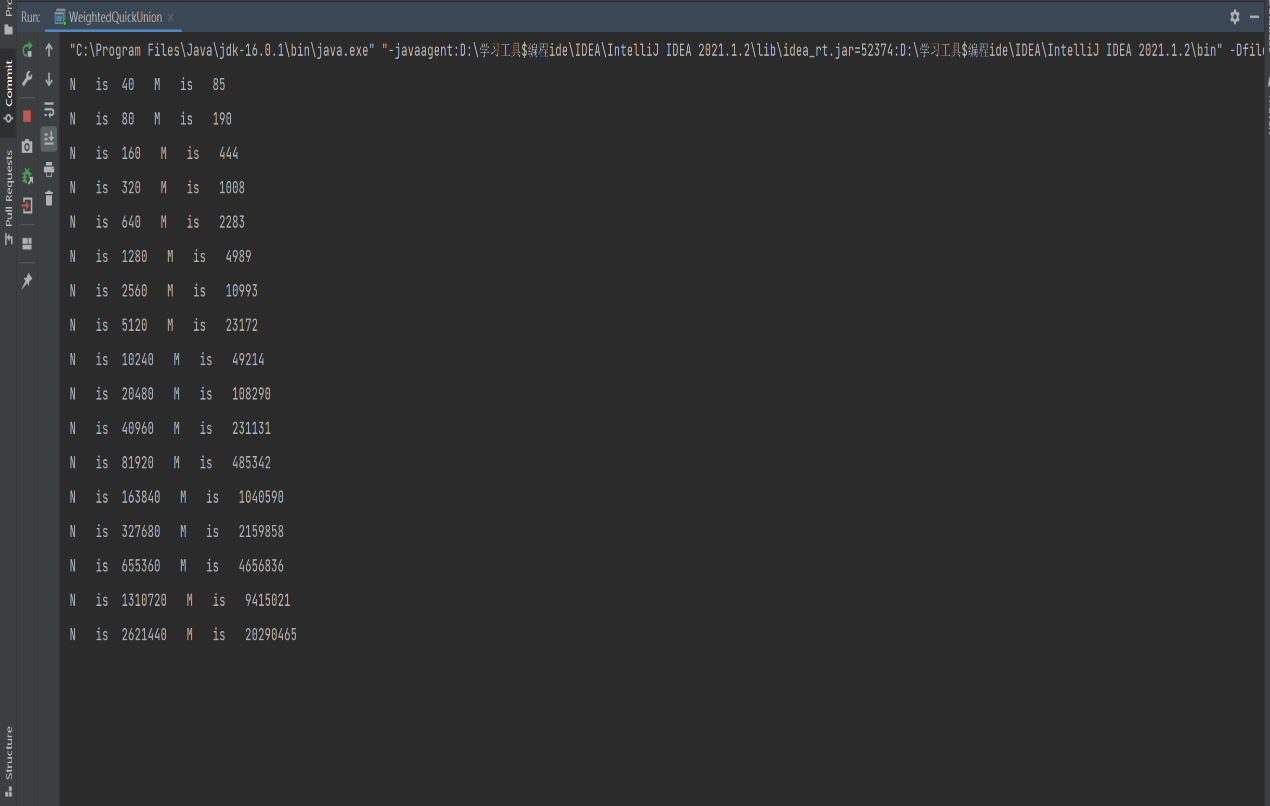
****

**Output:**

**(Experiment outcome)**



**(unit test)**

****