Program Structures and Algorithms
Spring 2023(SEC – 1)

NAME: PAWAN KUMAR KRISHNAN
NUID: 002743773

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).

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.

Relationship Conclusion:

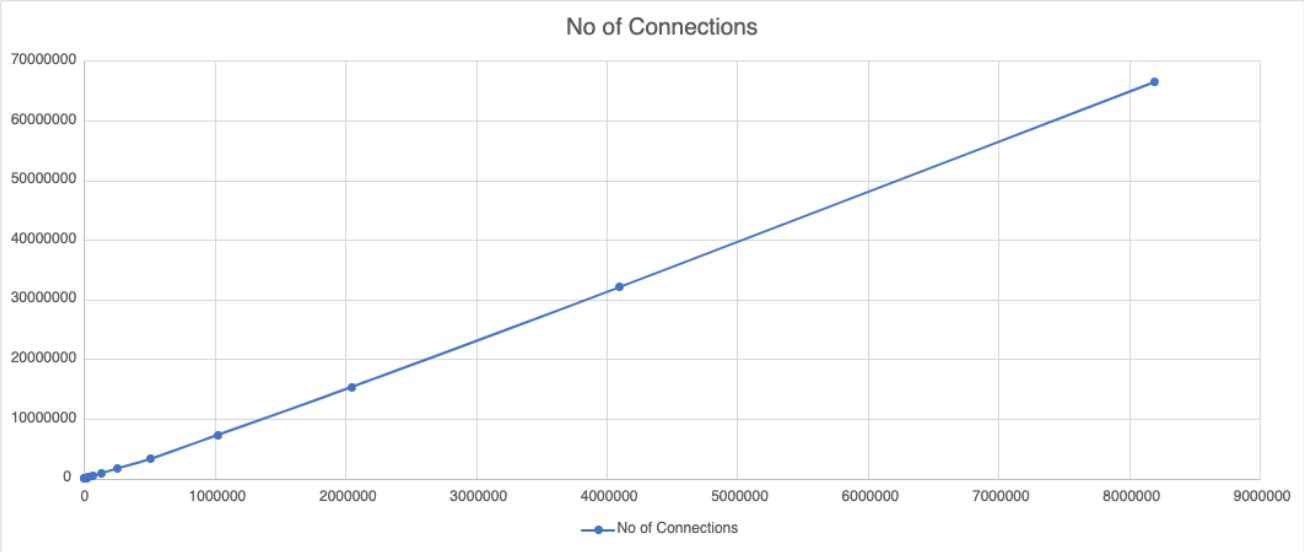
For the weighted union find with path compression assignment, the relationship is such that, as the number of sites increases(doubles in my case) the number of connections also approximately doubles. This shows that there is a linear relation ship between the number of sites and the number of connections.

Evidence to support that conclusion:

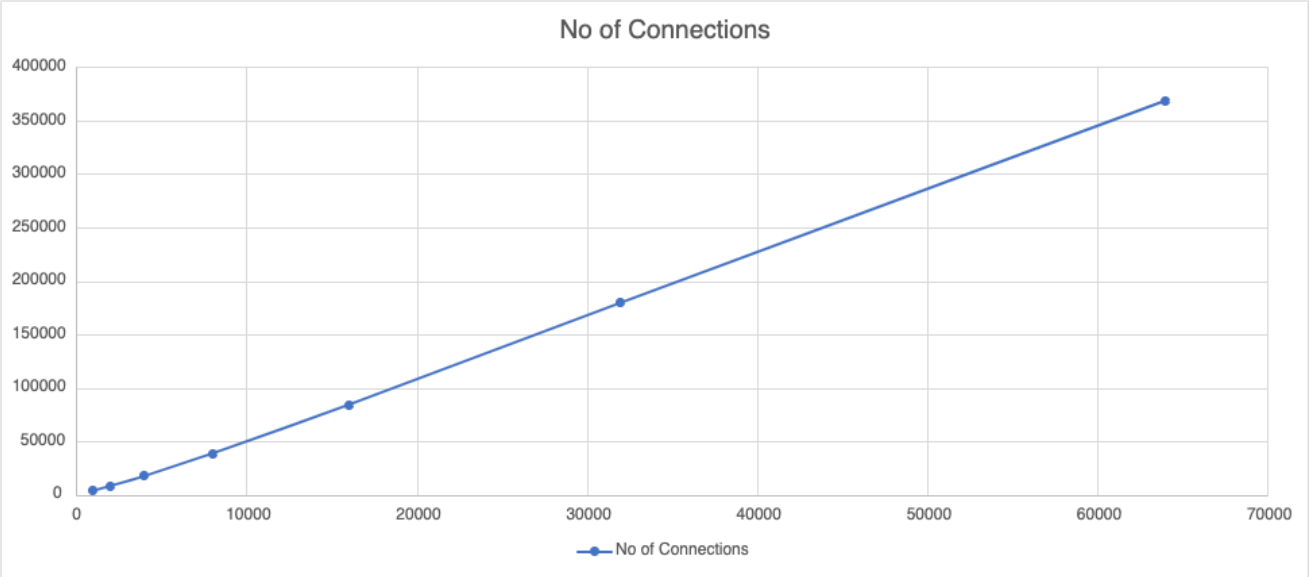
Number of sites vs Number of connection readings:

No of sites	No of Connections	log No of sites	log of No of Connections
1000	3717	3	3.570192561
2000	8165	3.301029996	3.911956189
4000	17378	3.602059991	4.239999793
8000	38591	3.903089987	4.586486032
16000	84103	4.204119983	4.924811488
32000	179626	4.505149978	5.254369199
64000	368644	4.806179974	5.56660717
128000	830954	5.10720997	5.919576983
256000	1679127	5.408239965	6.225083545
512000	3304349	5.709269961	6.519085911
1024000	7300378	6.010299957	6.863345348
2048000	15340683	6.311329952	7.185844696
4096000	32034132	6.612359948	7.505612961
8192000	66447151	6.913389944	7.822476365

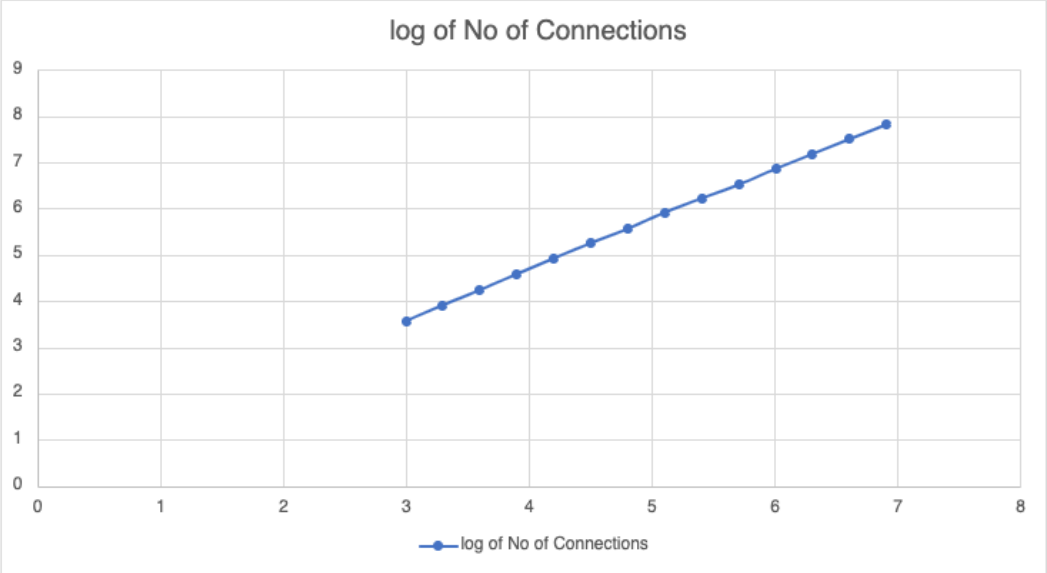
The graph shows the relationship of Number of sites vs Number of connection
Number of sites (x axis) vs Number of connection(y-axis)



Below is the same graph for lower number of sites,

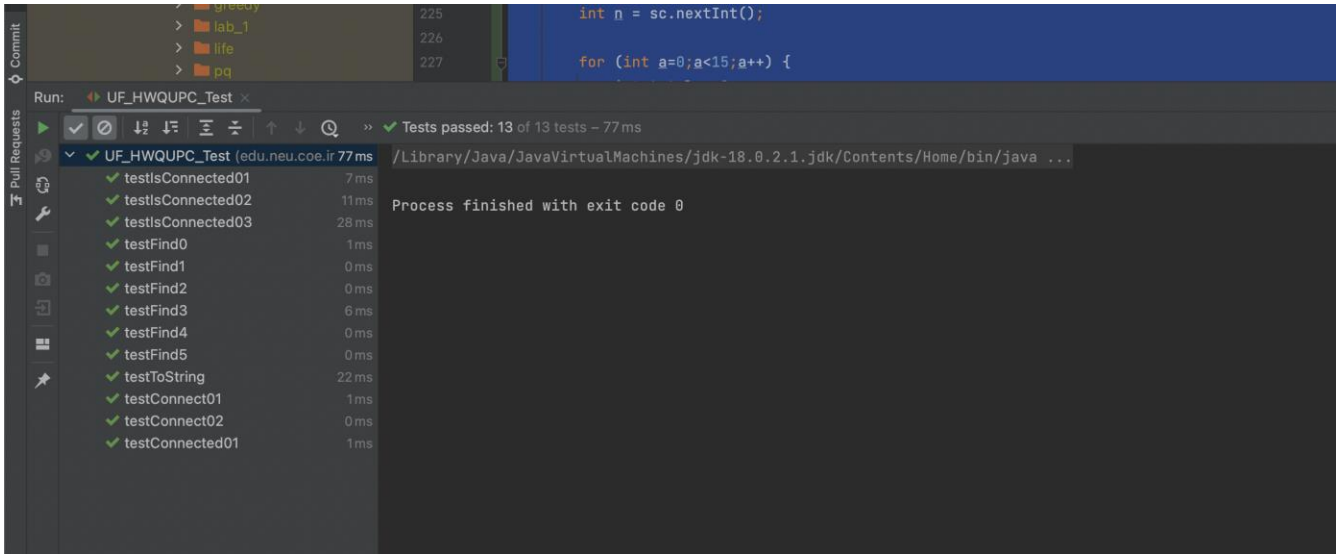


The log graph proves the linearity in the relationship.



Unit Test Screenshots:

Union Find with Path Compression


lab_1
life
pq
225
226
227
int n = sc.nextInt();
for (int a=0;a<15;a++) {
Run:
UF_HWQUPC_Test
Tests passed: 13 of 13 tests – 77 ms
UF_HWQUPC_Test (edu.neu.coe.ir 77 ms
testIsConnected01 7 ms
testIsConnected02 11 ms
testIsConnected03 28 ms
testFind0 1 ms
testFind1 0 ms
testFind2 0 ms
testFind3 6 ms
testFind4 0 ms
testFind5 0 ms
testToString 22 ms
testConnect01 1 ms
testConnect02 0 ms
testConnected01 1 ms
/Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java ...
Process finished with exit code 0
Commit
Pull Requests