

Program Structures & Algorithms Spring 2023

Assignment No. 4

TASK:

1. (a) Implement height-weighted Quick Union with Path Compression. Fill out the sections marked as "TO BE IMPLEMENTED" in the class UF_HWQUPC 1. (b) Check unit test cases all work for this class.
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).
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:

In order to minimize the number of components or objects from n to 1, or in other words, in order for all the pairs to be connected, the relationship between the number of objects (n) and the number of randomly produced pairs (m) can be expressed as follows:

We may conclude that after averaging the value of m over 50 executions of the program (for the same value of n), which was seen over n ranging from 1000 to 256000, is roughly equal to 1.22. (doubling)

As a result, the relationship can be expressed as $m \propto n * \log(n)$ (log taken to the base of 2).

$$m = c * n * \log(n)$$

$$\text{where } c = m/n * \log(n)$$

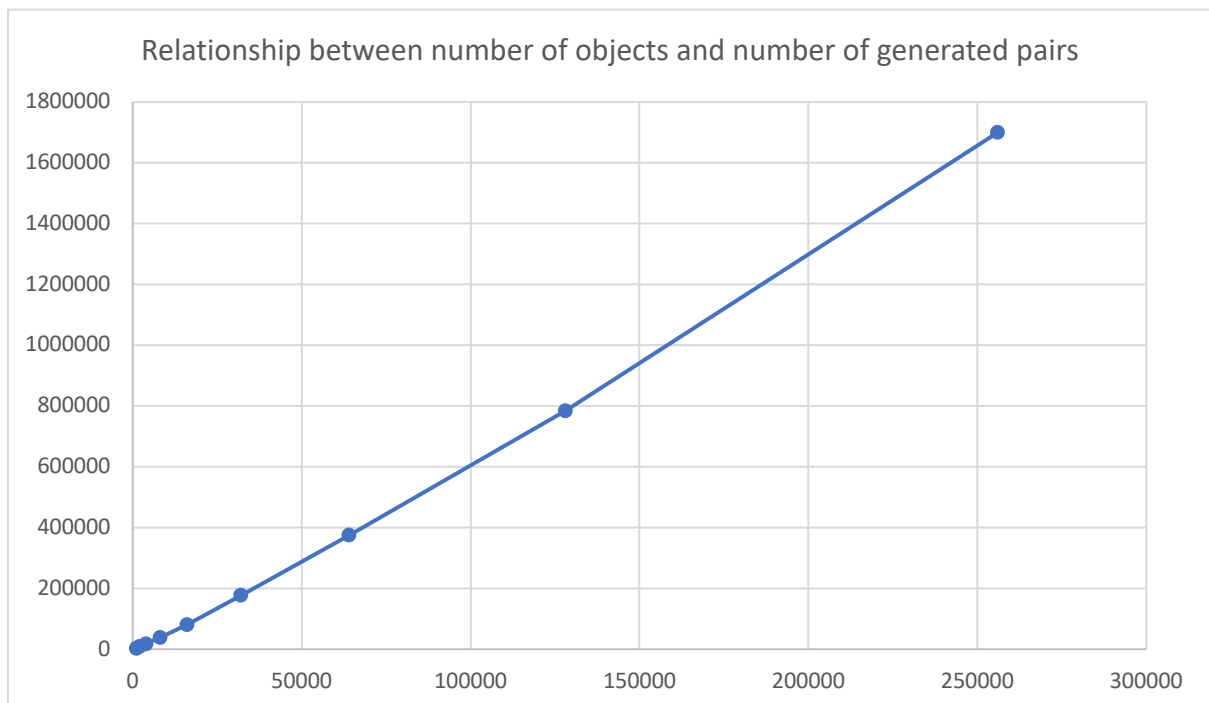
$$m \propto n * \log(n)$$

EVIDENCE TO SUPPORT THE CONCLUSION:

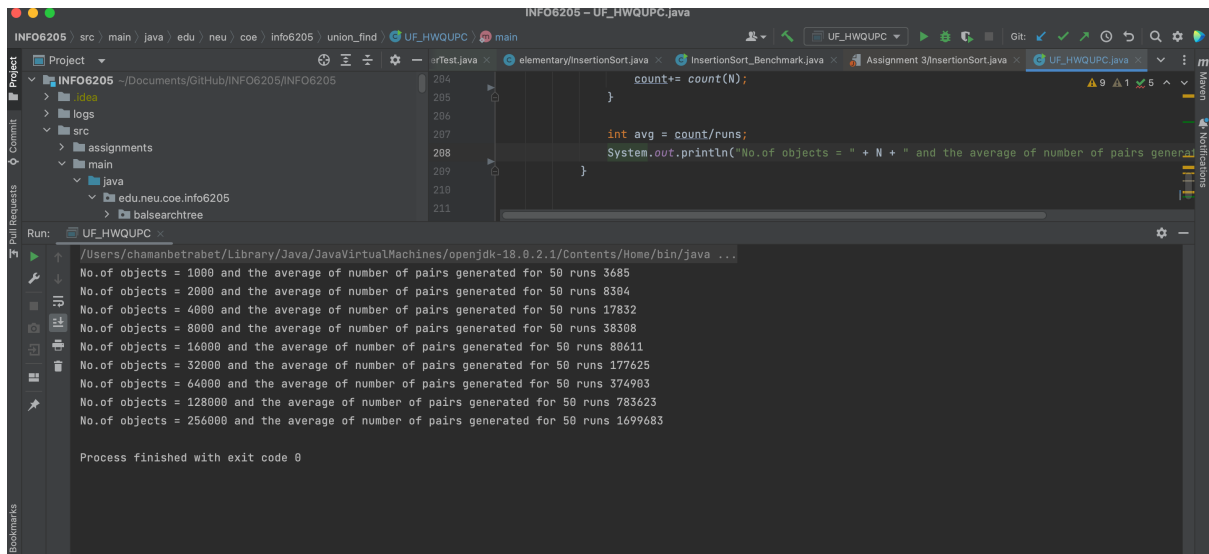
1. Have taken values for n ranging from 1000 to 256000 (doubling each time)
2. For each value of n, program is run for 50 times and the average value for m is noted
3. As explained above, constant c in the equation can roughly be approximated to 1.22 and therefore the relationship can be established as:

$$m = c * n * \log(n)$$

n (number of objects)	m (number of generated pairs) over an average of 50 runs	n log n	m/n log n
1000	3685	3000	1.22833333
2000	8304	6602.05999	1.25778924
4000	17832	14408.24	1.23762514
8000	38308	31224.7199	1.22684848
16000	80611	67265.9197	1.19839289
32000	177625	144164.799	1.23209688
64000	374903	307595.518	1.21881815
128000	783623	653722.876	1.19870824
256000	1699683	1384509.43	1.22764277

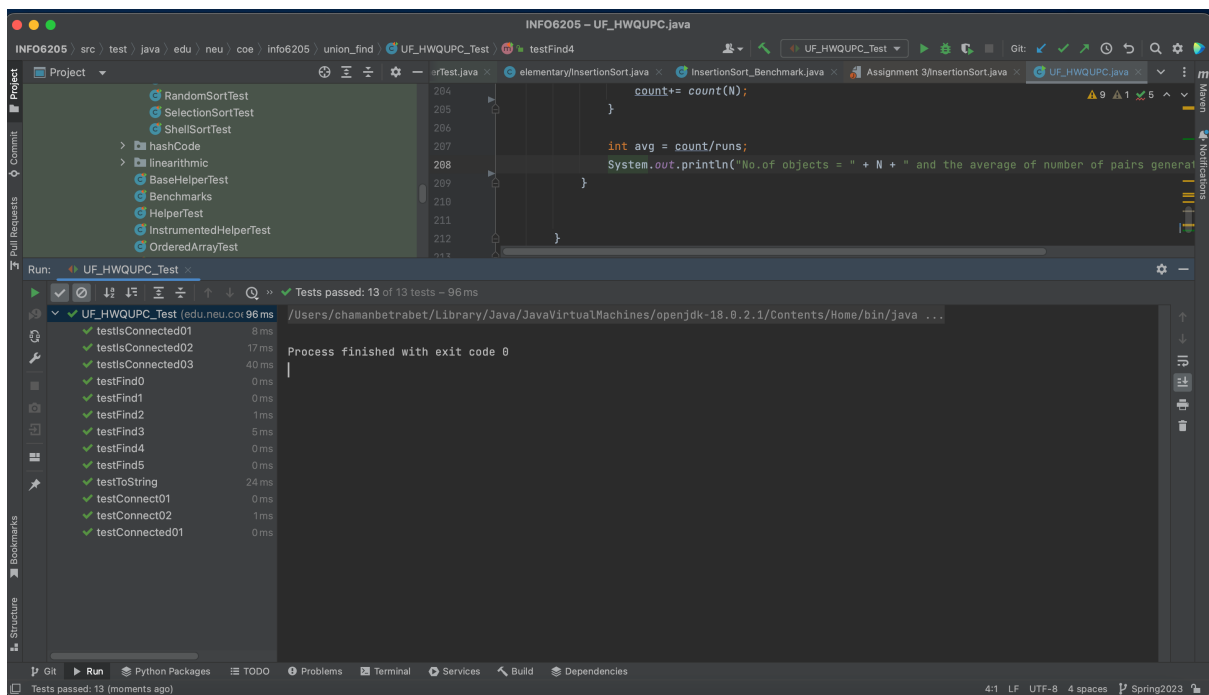


Output:



```
INFO6205 - UF_HWQUPC.java
INFO6205 src main java edu neu coe info6205 union_find UF_HWQUPC main
204 count+= count(N);
205 }
206
207 int avg = count/runs;
208 System.out.println("No. of objects = " + N + " and the average of number of pairs generated for 50 runs is " + avg);
209 }
210
211
Run: UF_HWQUPC
/Users/chamanbetrabet/Library/Java/JavaVirtualMachines/openjdk-18.0.2.1/Contents/Home/bin/java ...
No. of objects = 1000 and the average of number of pairs generated for 50 runs 3685
No. of objects = 2000 and the average of number of pairs generated for 50 runs 8304
No. of objects = 4000 and the average of number of pairs generated for 50 runs 17832
No. of objects = 8000 and the average of number of pairs generated for 50 runs 38308
No. of objects = 16000 and the average of number of pairs generated for 50 runs 80611
No. of objects = 32000 and the average of number of pairs generated for 50 runs 177625
No. of objects = 64000 and the average of number of pairs generated for 50 runs 374903
No. of objects = 128000 and the average of number of pairs generated for 50 runs 783623
No. of objects = 256000 and the average of number of pairs generated for 50 runs 1699683
Process finished with exit code 0
```

Unit test results:



```
INFO6205 - UF_HWQUPC.java
INFO6205 src test java edu neu coe info6205 union_find UF_HWQUPC_Test testFind4
RandomSortTest
SelectionSortTest
ShellSortTest
hashCode
linearithmic
BaseHelperTest
Benchmarks
HelperTest
InstrumentedHelperTest
OrderedArrayTest
204 count+= count(N);
205 }
206
207 int avg = count/runs;
208 System.out.println("No. of objects = " + N + " and the average of number of pairs generated for 50 runs is " + avg);
209 }
210
211
212
Run: UF_HWQUPC_Test
Tests passed: 13 of 13 tests - 96 ms
UF_HWQUPC_Test edu.neu.coe 96 ms
Process finished with exit code 0
testisConnected01 8 ms
testisConnected02 17 ms
testisConnected03 40 ms
testFind0 0 ms
testFind1 0 ms
testFind2 1 ms
testFind3 5 ms
testFind4 0 ms
testFind5 0 ms
testToString 24 ms
testConnect01 0 ms
testConnect02 1 ms
testConnected01 0 ms
```