

# Program Structures & Algorithms

Spring 2022

## Assignment No. 3

Name: Raj Mehta

(NUID): 001094914

- **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:

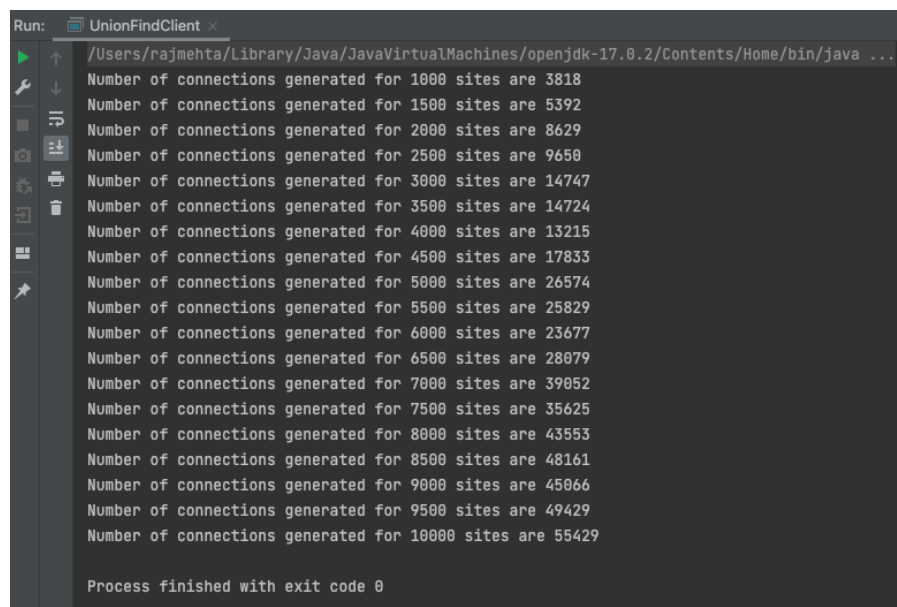
Using your implementation of UF\_HWQUPC, develop a UF ("union-find") client that takes an integer value  $n$  from the command line to determine the number of "sites." Then generates random pairs of integers between 0 and  $n-1$ , calling `connected()` to determine if they are connected and `union()` 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.

- **Output screenshot**

Output of Union Find Client:



```
Run: UnionFindClient x
/Users/rajmehta/Library/Java/JavaVirtualMachines/openjdk-17.0.2/Contents/Home/bin/java ...
Number of connections generated for 1000 sites are 3818
Number of connections generated for 1500 sites are 5392
Number of connections generated for 2000 sites are 8629
Number of connections generated for 2500 sites are 9650
Number of connections generated for 3000 sites are 14747
Number of connections generated for 3500 sites are 14724
Number of connections generated for 4000 sites are 13215
Number of connections generated for 4500 sites are 17833
Number of connections generated for 5000 sites are 26574
Number of connections generated for 5500 sites are 25829
Number of connections generated for 6000 sites are 23677
Number of connections generated for 6500 sites are 28079
Number of connections generated for 7000 sites are 39052
Number of connections generated for 7500 sites are 35625
Number of connections generated for 8000 sites are 43553
Number of connections generated for 8500 sites are 48161
Number of connections generated for 9000 sites are 45066
Number of connections generated for 9500 sites are 49429
Number of connections generated for 10000 sites are 55429

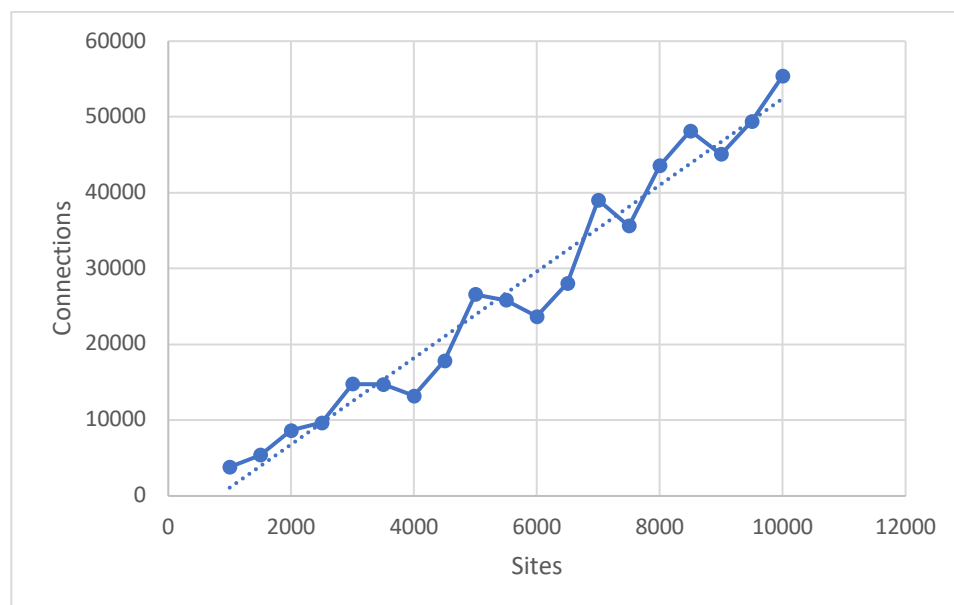
Process finished with exit code 0
```

- **Evidence / Graph**

As per the output above, the values and the relation between sites and connections are as follows:

Sites (n)	Connections (m)	Relation $n \cdot \log(n)$
1000	3818	3000
1500	5392	4764.136889
2000	8629	6602.059991
2500	9650	8494.850022
3000	14747	10431.36376
3500	14724	12404.23816
4000	13215	14408.23997
4500	17833	16439.45631
5000	26574	18494.85002
5500	25829	20571.99479
6000	23677	22668.9075
6500	28079	24783.93682
7000	39052	26915.68628
7500	35625	29062.95948
8000	43553	31224.7199
8500	48161	33400.06087
9000	45066	35588.18258
9500	49429	37788.37425
10000	55429	40000

The same data in graphical form is as follows:



- **Relationship Conclusion**

Based on the evidence shown above, the relation between the sites and connections is **approximately linear**.

Calculating the value of  $(n \cdot \log n)$  is approximately equal to the value of  $(m)$ .

That is, we can conclude that if the maximum height of node is  $\log(n)$  and unions done is  $n$ , then number of connections  $(m) = (n * \log n)$ .

- **Unit tests result**

