ANURAG PARLA (002127710)

Program Structures & Algorithms
Fall 2021

Assignment No. 03

<u>Task:</u>

1) a) To implement height-weighted Quick Union with Path Compression by fleshing out the class named UF_HWUQPC.
   b) Execute the test file named UF_HWUQPC_Test and make sure all the test cases pass with a green check mark indicating success.

2) a) Develop a Union find client that takes an integer value "n" from the command line to determine the number of sites.
   b) Generate random pairs of integers between 0 and n-1 by calling connected() to determine if they are connected and then call union() if they aren't and finally print the number of pairs generated.
   c) Develop a static method called count() which gets "n" as the argument and returns the number of connections and create a main() which will invoke the count() and prints the returned value.

3) Determine the relationship between the number of objects “n” and the number of pairs generated “m” in order to achieve the reduction of components from n to 1. In addition to this support the relationship deduced by appropriate graph and outputs from the terminal.

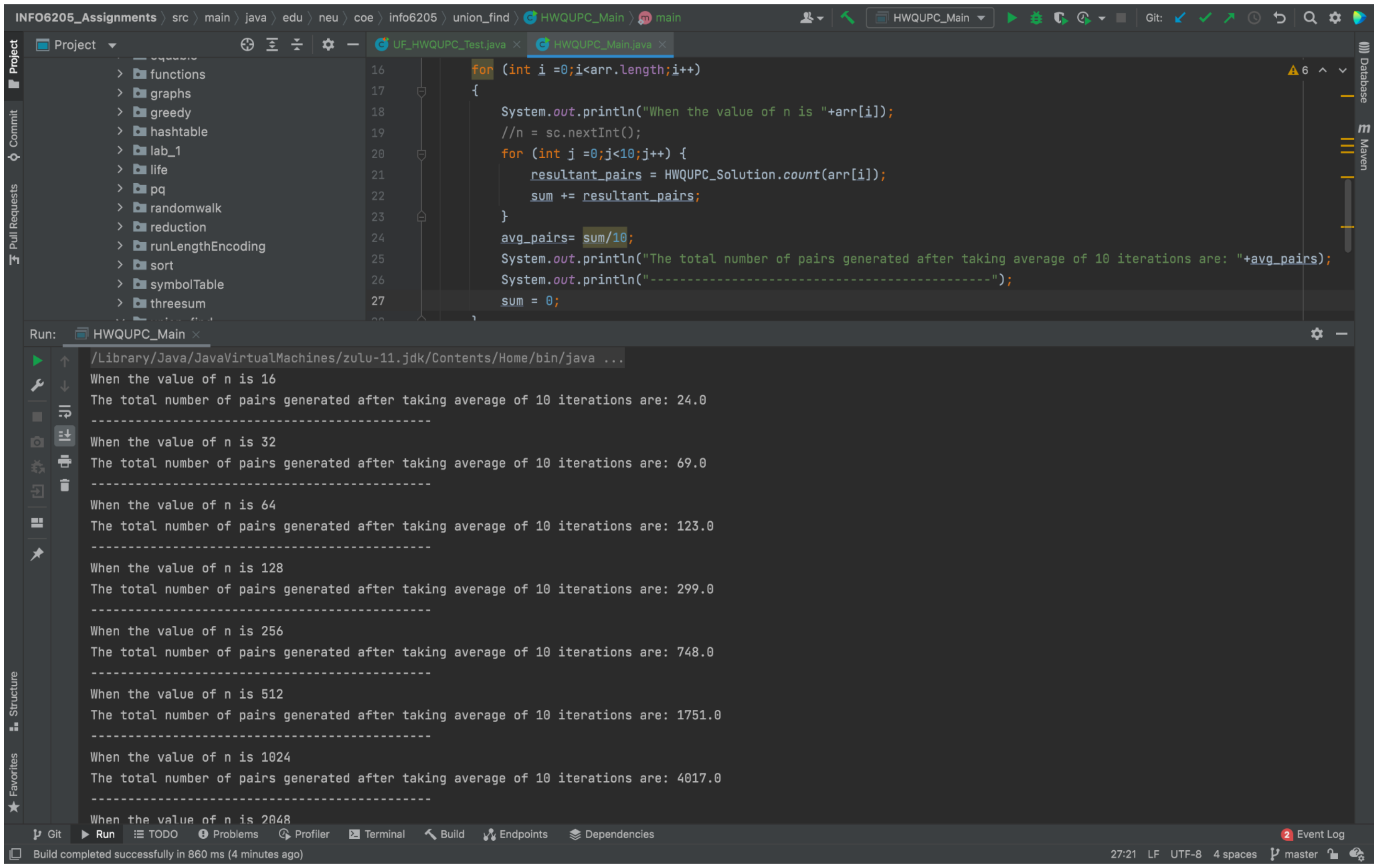
Relationship Conclusion for Task 3:
From the observation made by viewing the graph depicting the relationship between the number of objects “n” and the number of pairs generated “m” it is observed that the relationship is **linearithmic.**

Hence, **m ≅1/2 * nlogn**

Evidence to support the conclusion:

Screenshot of the average of number of pairs generated is displayed in the terminal for different values of n entered through command line:

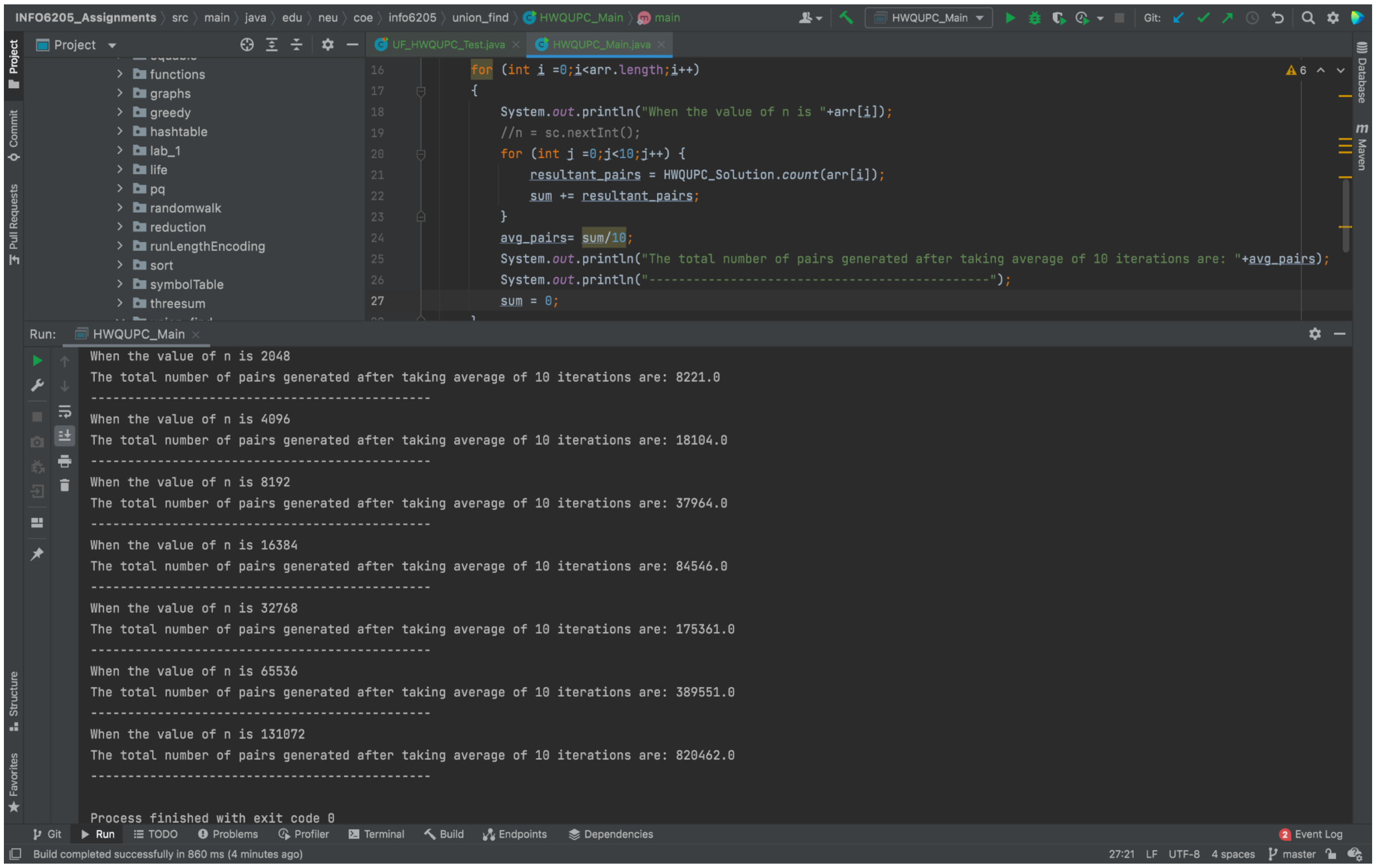
Here the total number of pairs generated is the average value which is obtained after iterating for 10 times for every value of “n”.



```
for (int i =0;i<arr.length;i++)
{
    System.out.println("When the value of n is "+arr[i]);
    //n = sc.nextInt();
    for (int j =0;j<10;j++) {
        resultant_pairs = HWQUPC_Solution.count(arr[i]);
        sum += resultant_pairs;
    }
    avg_pairs= sum/10;
    System.out.println("The total number of pairs generated after taking average of 10 iterations are: "+avg_pairs);
    System.out.println("---------------------------------------------");
    sum = 0;
```

```
/Library/Java/JavaVirtualMachines/zulu-11.jdk/Contents/Home/bin/java ...
When the value of n is 16
The total number of pairs generated after taking average of 10 iterations are: 24.0
---------------------------------------------
When the value of n is 32
The total number of pairs generated after taking average of 10 iterations are: 69.0
---------------------------------------------
When the value of n is 64
The total number of pairs generated after taking average of 10 iterations are: 123.0
---------------------------------------------
When the value of n is 128
The total number of pairs generated after taking average of 10 iterations are: 299.0
---------------------------------------------
When the value of n is 256
The total number of pairs generated after taking average of 10 iterations are: 748.0
---------------------------------------------
When the value of n is 512
The total number of pairs generated after taking average of 10 iterations are: 1751.0
---------------------------------------------
When the value of n is 1024
The total number of pairs generated after taking average of 10 iterations are: 4017.0
---------------------------------------------
When the value of n is 2048
```

Build completed successfully in 860 ms (4 minutes ago)



```java
        for (int i =0;i<arr.length;i++)
        {
            System.out.println("When the value of n is "+arr[i]);
            //n = sc.nextInt();
            for (int j =0;j<10;j++) {
                resultant_pairs = HWQUPC_Solution.count(arr[i]);
                sum += resultant_pairs;
            }
            avg_pairs= sum/10;
            System.out.println("The total number of pairs generated after taking average of 10 iterations are: "+avg_pairs);
            System.out.println("---------------------------------------------");
            sum = 0;
```

```
When the value of n is 2048
The total number of pairs generated after taking average of 10 iterations are: 8221.0
---------------------------------------------
When the value of n is 4096
The total number of pairs generated after taking average of 10 iterations are: 18104.0
---------------------------------------------
When the value of n is 8192
The total number of pairs generated after taking average of 10 iterations are: 37964.0
---------------------------------------------
When the value of n is 16384
The total number of pairs generated after taking average of 10 iterations are: 84546.0
---------------------------------------------
When the value of n is 32768
The total number of pairs generated after taking average of 10 iterations are: 175361.0
---------------------------------------------
When the value of n is 65536
The total number of pairs generated after taking average of 10 iterations are: 389551.0
---------------------------------------------
When the value of n is 131072
The total number of pairs generated after taking average of 10 iterations are: 820462.0
---------------------------------------------

Process finished with exit code 0
```

Build completed successfully in 860 ms (4 minutes ago)

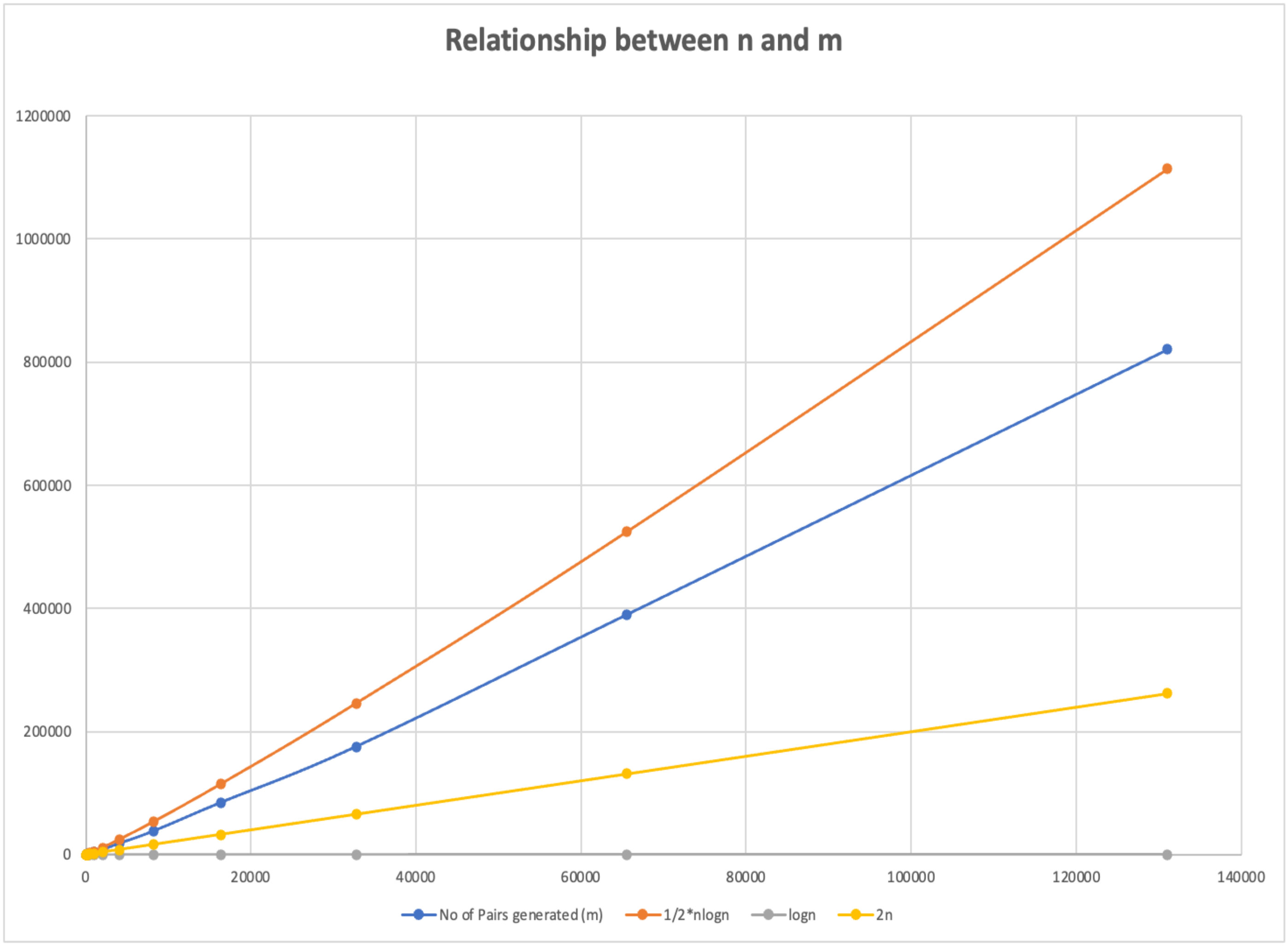
Table consisting of values of n, m, ½*NlogN, logN and 2N:

Here the value of “m” is the average value which is generated after iterating for 10 times for each value of “n”.

No of objects (n)	No of pairs generated (m)	½*(nlogn)	logn	2n
16	24	32	4	32
32	69	80	5	64
64	123	192	6	128

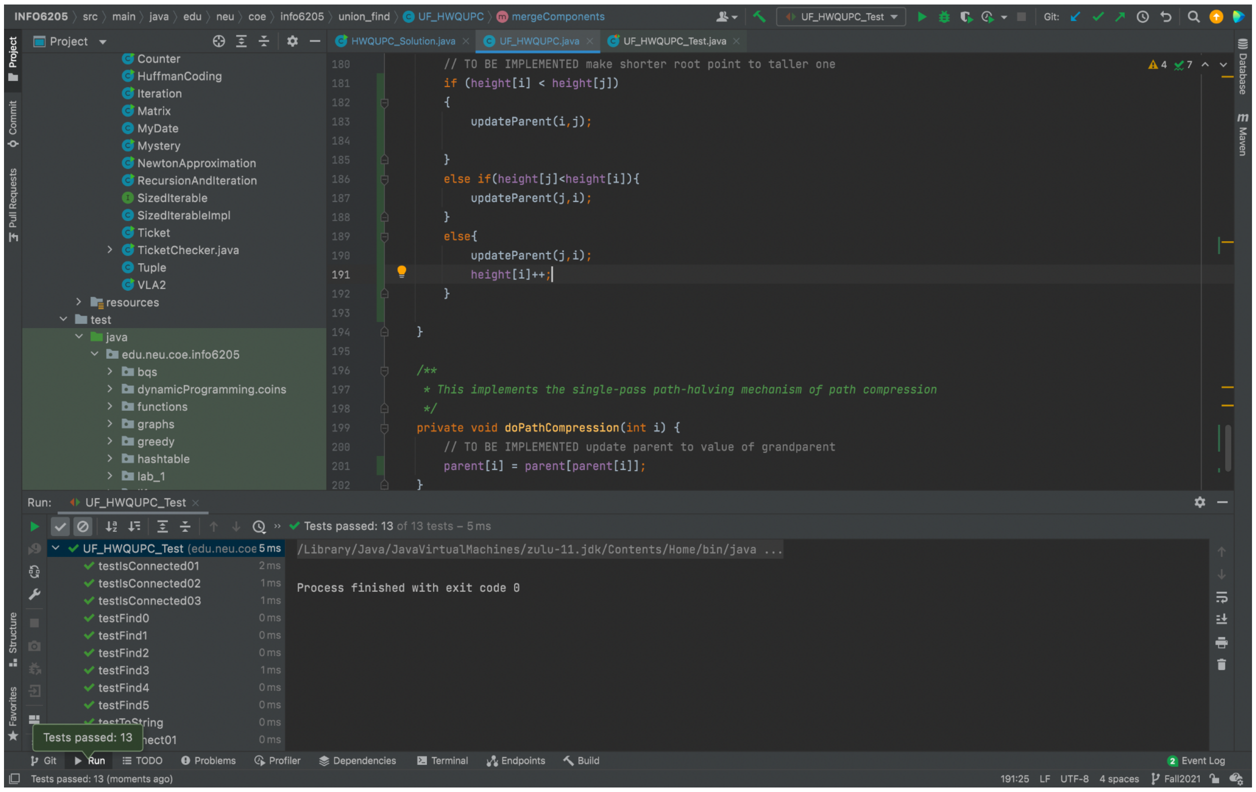
128	299	448	7	256
256	748	1024	8	512
512	1751	2304	9	1024
1024	4017	5120	10	2048
2048	8221	11264	11	4096
4096	18104	24576	12	8192
8192	37964	53248	13	16384
16384	84546	114688	14	32768
32768	175361	245760	15	65536
65536	389551	524288	16	131072
131072	820462	1114112	17	262144

Graphical relationship between the number of objects “n” and number of pairs generated “m”:



Unit tests result(Snapshot of successful unit test run):

Output of UF_HWQUPC_Test.java



```
        // TO BE IMPLEMENTED make shorter root point to taller one
        if (height[i] < height[j])
        {
            updateParent(i,j);

        }
        else if(height[j]<height[i]){
            updateParent(j,i);
        }
        else{
            updateParent(j,i);
            height[i]++;
        }

    }

    /**
     * This implements the single-pass path-halving mechanism of path compression
     */
    private void doPathCompression(int i) {
        // TO BE IMPLEMENTED update parent to value of grandparent
        parent[i] = parent[parent[i]];
    }
```

```
Tests passed: 13 of 13 tests – 5 ms

UF_HWQUPC_Test (edu.neu.coe 5 ms
    testIsConnected01     2 ms
    testIsConnected02     1 ms
    testIsConnected03     1 ms
    testFind0             0 ms
    testFind1             0 ms
    testFind2             0 ms
    testFind3             1 ms
    testFind4             0 ms
    testFind5             0 ms
    testToString          0 ms
    ...ect01              0 ms

/Library/Java/JavaVirtualMachines/zulu-11.jdk/Contents/Home/bin/java ...

Process finished with exit code 0

Tests passed: 13 (moments ago)
```