**Anjali Sajeevan (001563277)**

# Program Structures & Algorithms

# Spring 2021

# Assignment No. 3

- ## Task

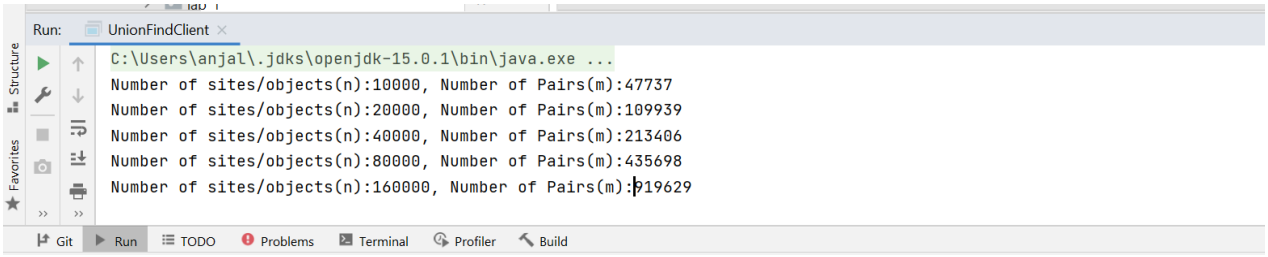
  - Implement height-weighted Quick Union with Path Compression. For this, you will flesh out the class UF_HWQUPC. All you have to do is to fill in the sections marked with // TO BE IMPLEMENTED ... // ...END IMPLEMENTATION. (b) Check that the unit tests for this class all work. You must show "green" test 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.

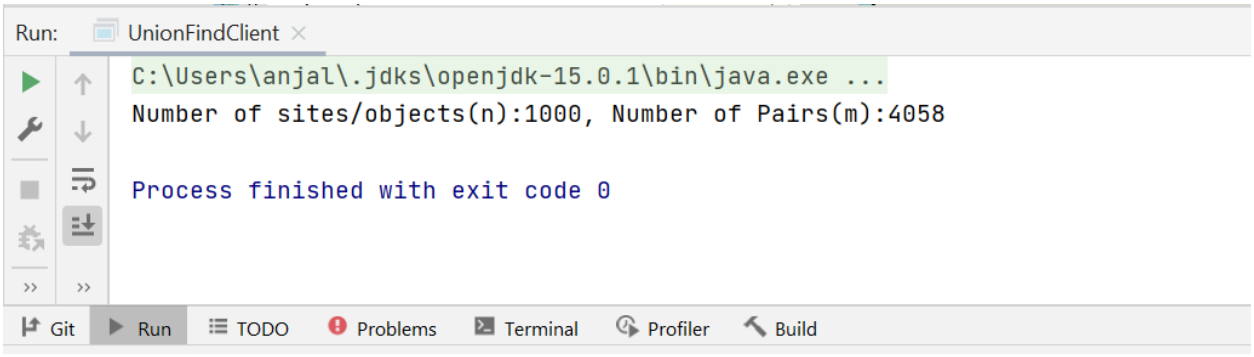
    If you prefer, you can create a main program that doesn't require any input and runs the experiment for a fixed set of n values. Show evidence of your run(s).

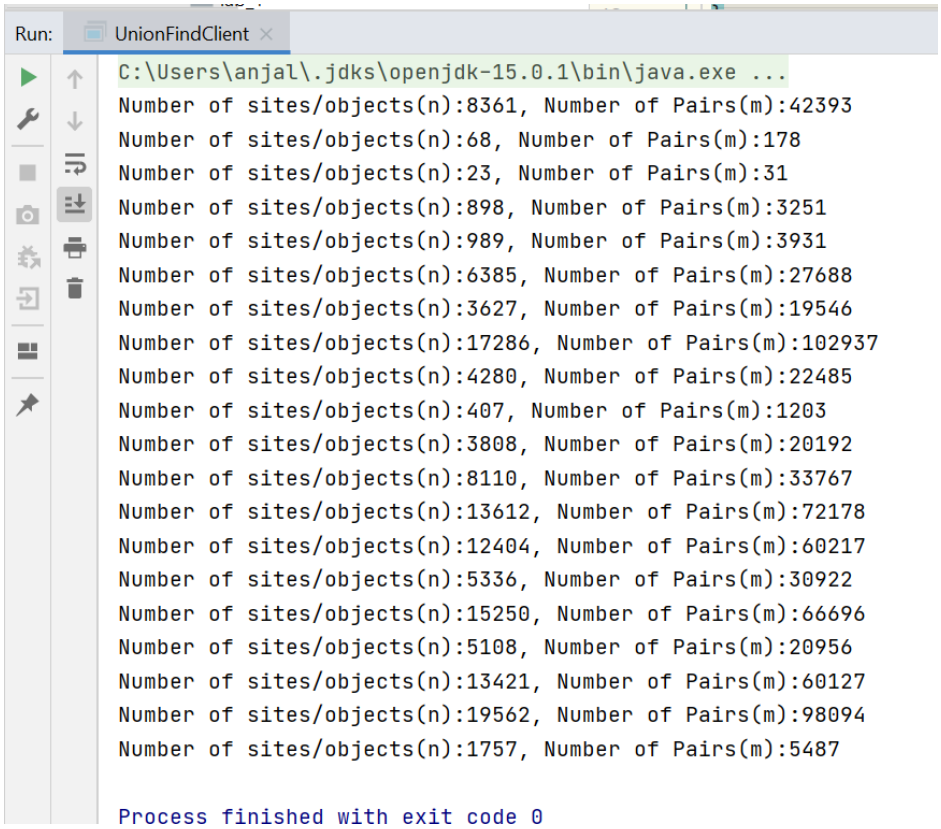
  - **Step 3:** Determine the relationship between the number of objects (*n*) and the number of pairs (*m*) generated to accomplish this (i.e., to reduce the number of components from *n* to 1). Justify your conclusion.

- ## Solution

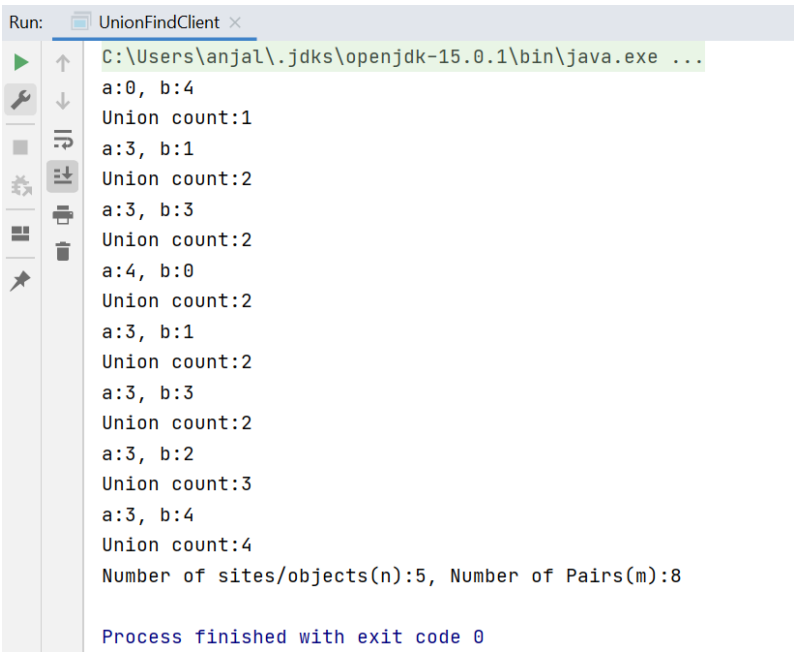
The methods doPathCompression(int i), mergeComponents(int i, int j) and find(int p) was implemented in the UF_HWQUPC class. All the unit tests of UF_HWQUPC_Test has passed. The UnionFindClient class was implemented and the relation between the number of objects (*n*) and the number of pairs (*m*) generated was determined. The below figures show the number of pairs generated for different values of n.




This figure below shows the chosen random points of a and b, the number of unions and the number of pairs(m) for n number of sites are shown. The union count is the number of unions made, for 5 number of sites the unions was generated to be 4. It was checked for various values of n and the relationship was determined as n-1.




- ## Relationship Conclusion:

The relationship between the number of objects/sites(n) and the number of pairs(m) can be concluded from the results that its higher than n but lower than nlogn, the values obtained are very close to $\frac{1}{3}\boldsymbol{nlogn}$ . i.e., number of pairs generated is ~$\frac{1}{3}\boldsymbol{nlogn} = c * \boldsymbol{nlogn,}$ where c is a constant and logarithm is to the base 2. When natural logarithm is considered then the number of pairs generated is ~$\frac{1}{2}\boldsymbol{nln\ n.}$

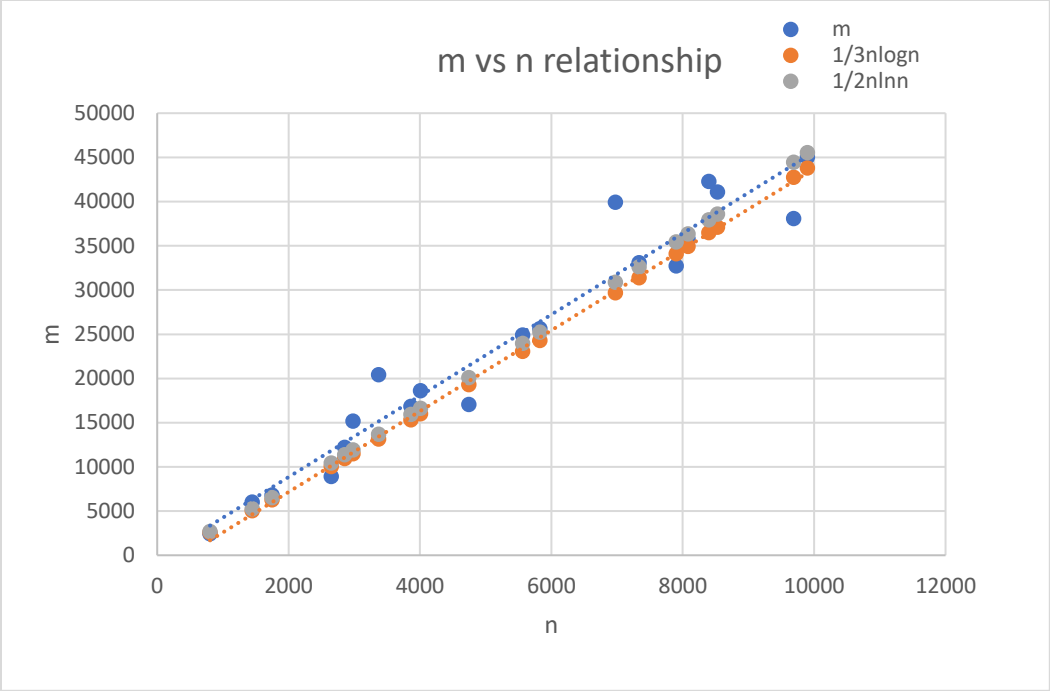
- ## Evidence to support the conclusion and graphical representation:

The below results are obtained for 20 random values of n (number of sites). The graphical representation is plotted below. The blue represents the values of m (number of pairs), the orange represents 1/3nlogn values and grey represents 1/2nln n values.

| n | m | base | logn | nlogn | 1/3nlogn | ln n | n ln n | 1/2n ln n |
|---|---|---|---|---|---|---|---|---|
| 804 | 2418 | 2 | 9.651052 | 7759.446 | 2586.482 | 6.689599 | 5378.438 | 2689.219 |
| 1444 | 6021 | 2 | 10.49586 | 15156.01 | 5052.005 | 7.275172 | 10505.35 | 5252.674 |
| 1749 | 6815 | 2 | 10.77231 | 18840.78 | 6280.259 | 7.466799 | 13059.43 | 6529.716 |
| 2651 | 8926 | 2 | 11.37232 | 30148.02 | 10049.34 | 7.882692 | 20897.02 | 10448.51 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 2855 | 12209 | 2 | 11.47928 | 32773.33 | 10924.44 | 7.956827 | 22716.74 | 11358.37 |
| 2981 | 15161 | 2 | 11.54158 | 34405.45 | 11468.48 | 8.000014 | 23848.04 | 11924.02 |
| 3368 | 20436 | 2 | 11.71768 | 39465.13 | 13155.04 | 8.122074 | 27355.15 | 13677.57 |
| 3860 | 16839 | 2 | 11.91439 | 45989.53 | 15329.84 | 8.258422 | 31877.51 | 15938.76 |
| 4008 | 18625 | 2 | 11.96867 | 47970.42 | 15990.14 | 8.296048 | 33250.56 | 16625.28 |
| 4745 | 17043 | 2 | 12.21219 | 57946.85 | 19315.62 | 8.464847 | 40165.7 | 20082.85 |
| 5560 | 24902 | 2 | 12.44087 | 69171.23 | 23057.08 | 8.623353 | 47945.84 | 23972.92 |
| 5825 | 25646 | 2 | 12.50804 | 72859.35 | 24286.45 | 8.669914 | 50502.25 | 25251.13 |
| 6976 | 39947 | 2 | 12.76818 | 89070.85 | 29690.28 | 8.850231 | 61739.21 | 30869.61 |
| 7333 | 33089 | 2 | 12.84019 | 94157.1 | 31385.7 | 8.90014 | 65264.73 | 32632.36 |
| 7899 | 32732 | 2 | 12.94745 | 102271.9 | 34090.65 | 8.974491 | 70889.51 | 35444.75 |
| 8078 | 35775 | 2 | 12.97978 | 104850.7 | 34950.23 | 8.9969 | 72676.95 | 36338.48 |
| 8397 | 42267 | 2 | 13.03566 | 109460.4 | 36486.81 | 9.03563 | 75872.18 | 37936.09 |
| 8528 | 41088 | 2 | 13.05799 | 111358.6 | 37119.52 | 9.05111 | 77187.87 | 38593.93 |
| 9687 | 38089 | 2 | 13.24183 | 128273.6 | 42757.88 | 9.17854 | 88912.52 | 44456.26 |
| 9897 | 45038 | 2 | 13.27278 | 131360.7 | 43786.89 | 9.199987 | 91052.27 | 45526.14 |

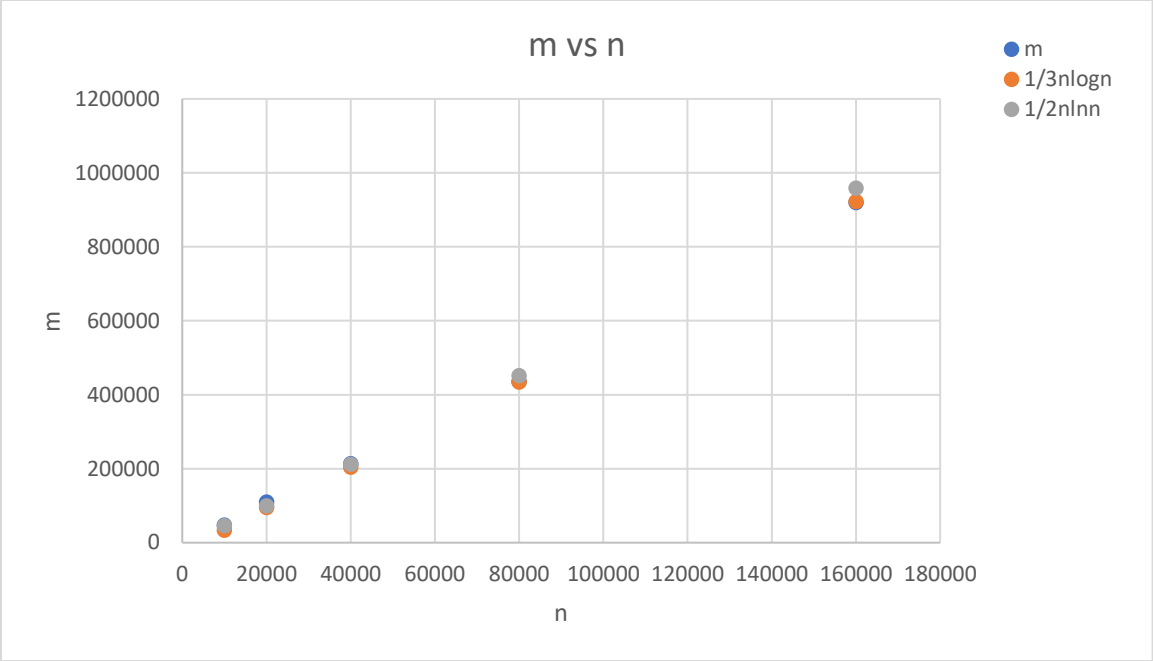
| n | m | 1/3nlogn | 1/2nlnn |
|---|---|---|---|
| 804 | 2418 | 2586.482 | 2689.219 |
| 1444 | 6021 | 5052.005 | 5252.674 |
| 1749 | 6815 | 6280.259 | 6529.716 |
| 2651 | 8926 | 10049.34 | 10448.51 |
| 2855 | 12209 | 10924.44 | 11358.37 |
| 2981 | 15161 | 11468.48 | 11924.02 |
| 3368 | 20436 | 13155.04 | 13677.57 |
| 3860 | 16839 | 15329.84 | 15938.76 |
| 4008 | 18625 | 15990.14 | 16625.28 |
| 4745 | 17043 | 19315.62 | 20082.85 |
| 5560 | 24902 | 23057.08 | 23972.92 |
| 5825 | 25646 | 24286.45 | 25251.13 |
| 6976 | 39947 | 29690.28 | 30869.61 |
| 7333 | 33089 | 31385.7 | 32632.36 |
| 7899 | 32732 | 34090.65 | 35444.75 |
| 8078 | 35775 | 34950.23 | 36338.48 |
| 8397 | 42267 | 36486.81 | 37936.09 |
| 8528 | 41088 | 37119.52 | 38593.93 |
| 9687 | 38089 | 42757.88 | 44456.26 |
| 9897 | 45038 | 43786.89 | 45526.14 |




When the values of n is increased the below results are obtained.

| n | m | base | logn | nlogn | 1/3nlogn | ln n | nln n | 1/2nlnn |
|---|---|---|---|---|---|---|---|---|
| 10000 | 47737 | 2 | 13.28771 | 132877.1 | 33219.28 | 9.21034 | 92103.4 | 46051.7 |
| 20000 | 109939 | 2 | 14.28771 | 285754.2 | 95251.42 | 9.903488 | 198069.8 | 99034.88 |
| 40000 | 213406 | 2 | 15.28771 | 611508.5 | 203836.2 | 10.59663 | 423865.4 | 211932.7 |
| 80000 | 435698 | 2 | 16.28771 | 1303017 | 434339 | 11.28978 | 903182.6 | 451591.3 |
| 160000 | 919629 | 2 | 17.28771 | 2766034 | 922011.3 | 11.98293 | 1917269 | 958634.3 |

| n | m | 1/3nlogn | 1/2nlnn |
|---|---|---|---|
| 10000 | 47737 | 33219.28 | 46051.7 |
| 20000 | 109939 | 95251.42 | 99034.88 |
| 40000 | 213406 | 203836.2 | 211932.7 |
| 80000 | 435698 | 434339 | 451591.3 |
| 160000 | 919629 | 922011.3 | 958634.3 |




As per the above graph it can be seen that the values obtained for the number of pairs-m ( blue) are very close or overlapping 1/3nlogn(orange) and 1/2nlnn(grey). Hence this provides the evidence that number of pairs generated is ~ $\frac{1}{2}\boldsymbol{nln\,n}$ when natural logarithm is considered and it is ~ $\frac{1}{3}\boldsymbol{nlog\,n}$ when logarithm to the base 2 is considered.

- # Unit tests result:

1. UF-HWQUPC_Test– Test case result