Program Structures and Algorithms
Spring 2023(SEC –1)

NAME: Manish Shivaprasad
NUID:002766390

Task

Step 1) Implement height-weighted Quick Union with Path Compression with all the testcases
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."
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).

Relationship Conclusion

The relationship between the number of objects (*n*) and the number of pairs (*m*) generated to reduce the number of components from *n* to 1 is

$$m=0.5*n*\ln(n).$$

where m=number of pairs, n=number of objects

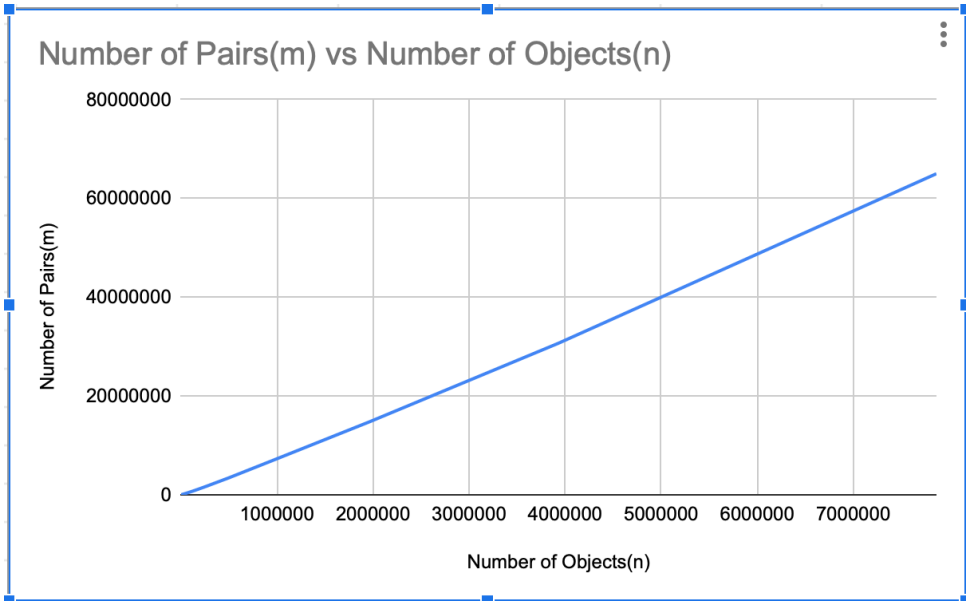
Evidence to support that conclusion

Using doubling method for N(Number of objects) with an initial value we can calculate Number of pairs(m).Below is the spreadsheet of data obtained.

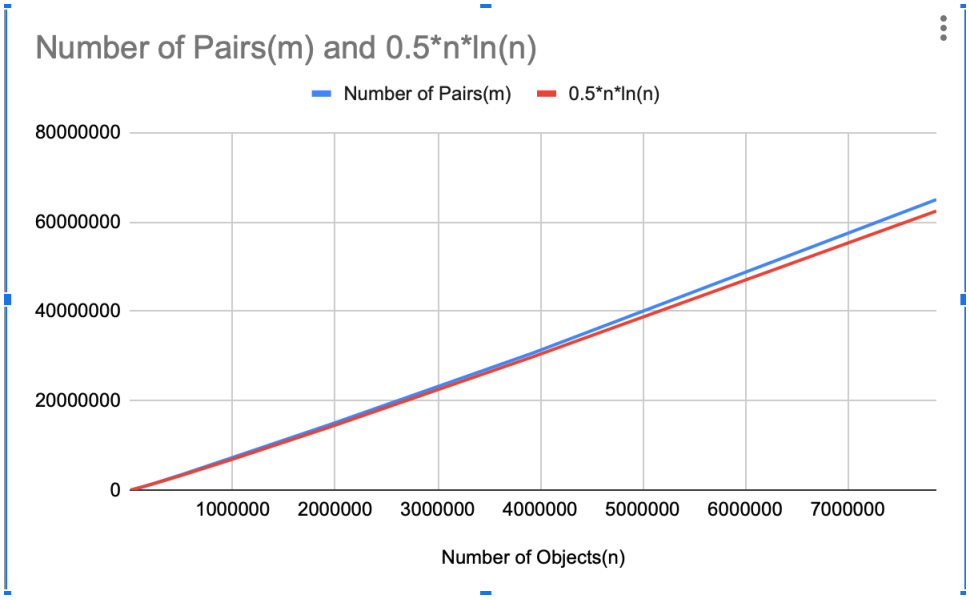
Number of Objects(n)	Number of Pairs(m)	0.5*n*ln(n)
30	66.24	51.01796072
60	142.12	122.8303369
120	332.52	287.2495046
240	758.96	657.6766708
480	1645.2	1481.708665
960	3426.88	3296.127977
1920	7786.16	7257.677246
3840	16659.72	15846.19708
7680	37748.36	34354.07933
15360	76140.04	74031.52901
30720	172075.12	158709.7987
61440	355034.28	338713.0788
122880	736906.36	720013.1204
245760	1585881.32	1525200.166
491520	3377903.12	3220748.184
983040	7168016.12	6782192.07
1966080	1.48E+07	14245775.54
3932160	3.07E+07	29854333.9
7864320	6.50E+07	62434233.41

We can see that number of pairs (m) is almost(approximately) equal to 0.5*n*ln(n)
Below are the graph representations of the above data obtained.

Graphical Representation



The graph above represents the relationship between the number of objects and the number of pairs calculated for those objects. The X-axis displays the number of objects, and the Y-axis shows the corresponding number of pairs.



The above graph compares the values of the number of pairs obtained and calculated using a relationship. It can be observed that both follow a similar trend.

Unit Test Screenshots

```java
package edu.neu.coe.info6205.union_find;

import edu.neu.coe.info6205.util.PrivateMethodTester;

public class UF_HWQUPC_Test {

    @Test
    public void testToString() {
        Connections h = new UF_HWQUPC(2);
        assertEquals("UF_HWQUPC:\n" +
                "  count: 2\n" +
                "  path compression? true\n" +
                "  parents: [0, 1]\n" +
                "  heights: [1, 1]", h.toString());
    }

    /**
     *
     */
    @Test
    public void testIsConnected01() {
        Connections h = new UF_HWQUPC(2);
        assertFalse(h.isConnected(0, 1));
    }

    /**
     *
```