Program Structures and Algorithms
Fall 2022(SEC 06)

NAME: Mahathi Siddavatam
NUID: 002198134

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.

NOTE: although I'm not going to tell you in advance what the relationship is, I can assure you that it is a *simple* relationship.

Don't forget to follow the submission guidelines. And to use sufficient (and sufficiently large) different values of n.

Relationship Conclusion: The relationship between the number of objects (n) and number of pairs generated is approximately:

number of pairs generated = n + nlog(base 10) n. Many of these connections, while redundant, still count in the total.

Evidence to support that conclusion:

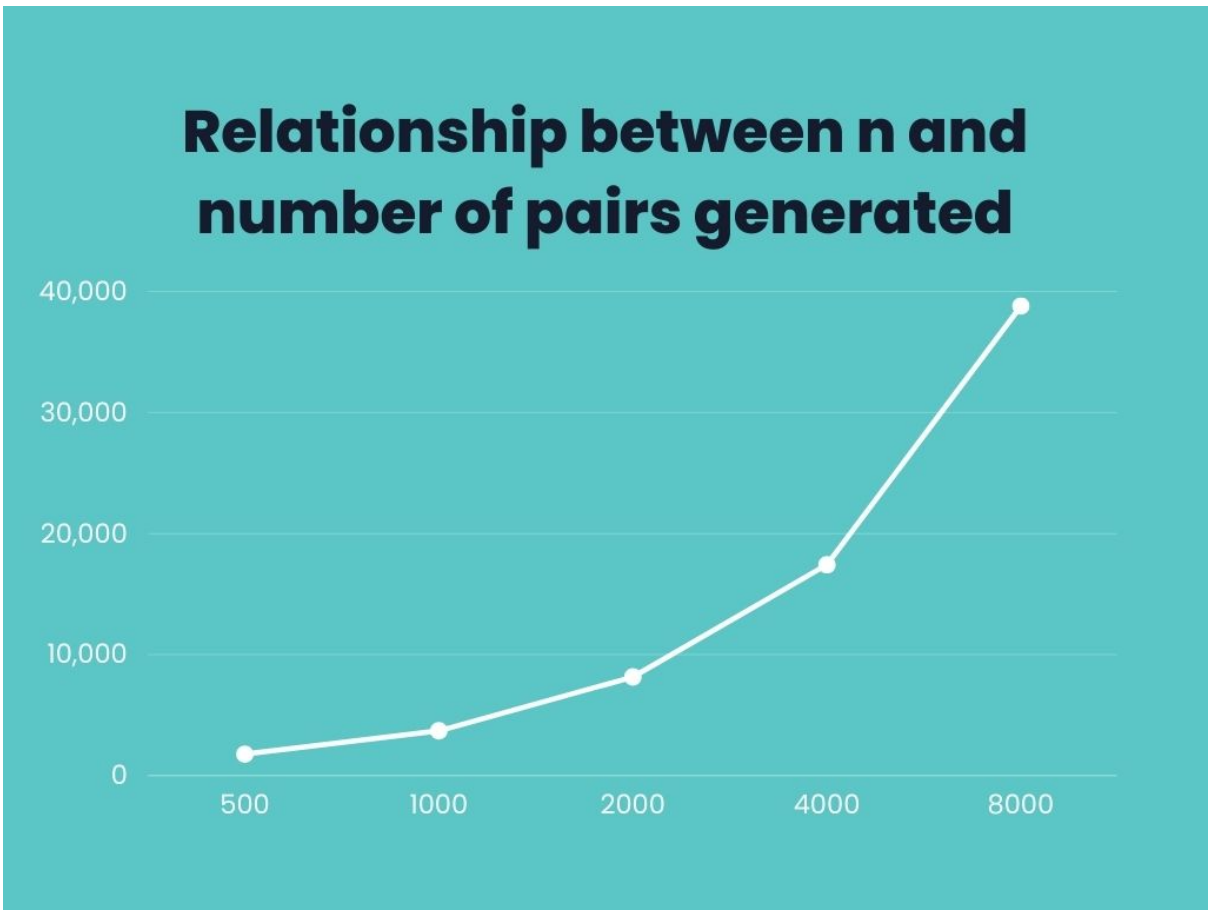
UF_HWQUPC.java | UF_HWQUPC_Test.java

```java
            i=parent[i];

        }

    }

    public static int count(int n) {
        UF_HWQUPC h = new UF_HWQUPC(n,true);
        int connections=0;

        Random rand = new Random();
        //System.out.println("Max pairs= "+MAX_PAIRS);

        while(h.components()>1) {

            int num1 = rand.nextInt(n);
            int num2 = rand.nextInt(n);

            h.connect(num1,num2);
            connections++;



        }
```

Console

```
UF_HWQUPC [Java Application] /Users/user/Library/Java/JavaVirtualMachines/openjdk-17.0.2/Contents/Home/bin/java  (Nov 8, 2022, 11:14:00 PM)
Enter n
500
Number of connections after running value 100 times = 1748
Enter n
1000
Number of connections after running value 100 times = 3675
Enter n
2000
Number of connections after running value 100 times = 8127
Enter n
4000
Number of connections after running value 100 times = 17406
Enter n
8000
Number of connections after running value 100 times = 38181
Enter n
```

Graphical Representation:



Unit Test Screenshots: