Sribalaji Prabhakar (001517537)

Program Structures & Algorithms

Spring 2021

Assignment No. 3

Task:

Our task is to Implement a quick union with path compression by completing the three methods (find, Merge and doCompression) in UF_HWQUPC.java. Once our test cases pass, we need to generate random pairs of integers(m) between 0 to n-1 until our component becomes 1. After that we need to deduce the relationship between N and M.

Part 1:

Added the following logic and all test cases were passed.

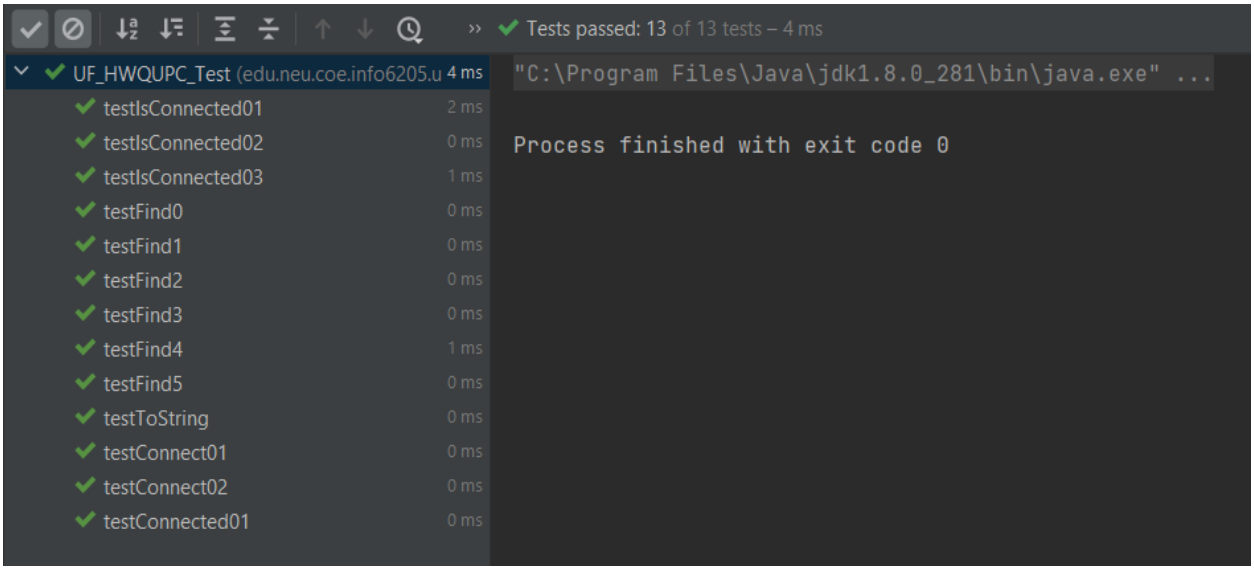
```
public int find(int p) {

    validate(p);
    int root = p;
    // TO BE IMPLEMENTED
    while (root != parent[root])
        root = parent[root];
    if (!pathCompression)
        return root;
    doPathCompression(p);
        return root;
}
```

```java
private void mergeComponents(int i, int j) {
    // TO BE IMPLEMENTED make shorter root point to taller one
    int rootP = find(i);
    int rootQ = find(j);
    if (rootP == rootQ) return;

    // make smaller root point to larger one
    if (height[rootP] < height[rootQ]) {
        parent[rootP] = rootQ;
    }
    else if(height[rootP] == height[rootQ]){
        parent[rootQ] = rootP;
        height[rootP]++;
    }
    else {
        parent[rootQ] = rootP;
    }
}
```

```java
private void doPathCompression(int i) {
    // TO BE IMPLEMENTED update parent to value of grandparent
    parent[i] = parent[parent[i]];
}
```



Task 2:

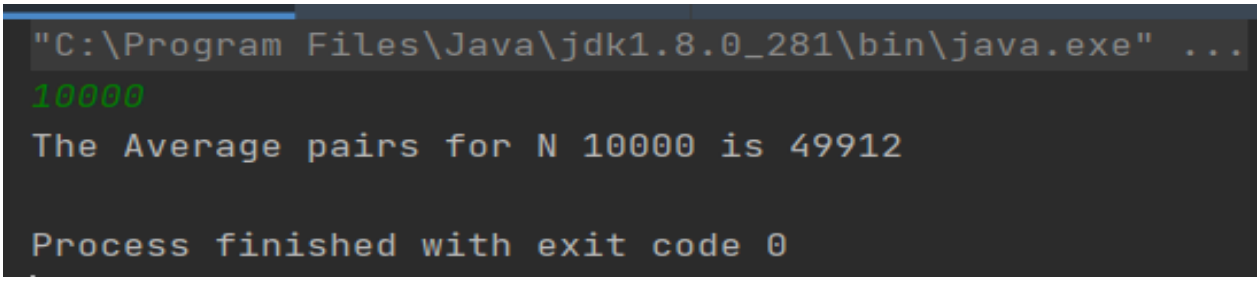
Created a main and generatePairs methods in UF_HWQUPC.java and implemented the below logic to generate and count random pairs until the components becomes 1. Ran the random pairs method 5 times and calculated the average M value for the accuracy.

1. For N = 10000 then M is approximately 49912
2. For N = 20000 then M is approximately 116078
3. For N = 40000 then M is approximately 222290

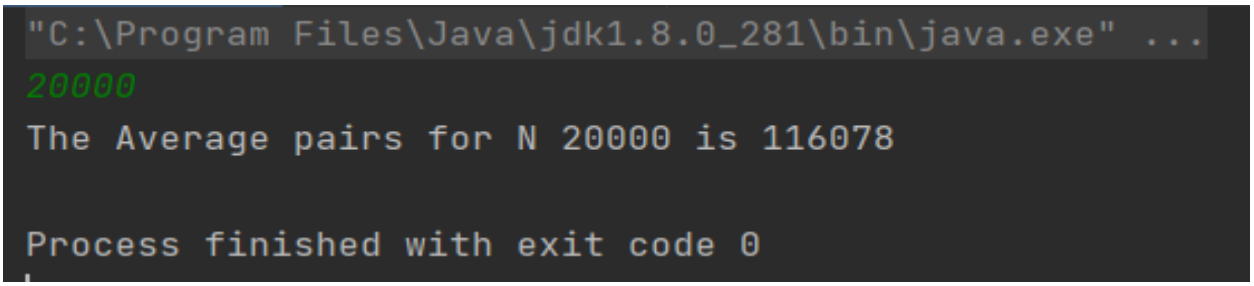
```
public static int generatePairs(int N){
    UF_HWQUPC uf_hwqupc = new UF_HWQUPC(N,  pathCompression: true);
    Random rand = new Random();
    int m = 0;
    while(uf_hwqupc.components() != 1) {
        int r1 = rand.nextInt(N);
        int r2 = rand.nextInt(N);
        uf_hwqupc.connect(r1,r2);
        m++;
    }
    return m;
}

public static void main(String args[]){
    Scanner in = new Scanner(System.in);
    int N = in.nextInt();
    Deque<Integer> average = new ArrayDeque<>();
    for(int i =1 ; i <= 5 ; i++)
        average.push(generatePairs(N));
    int avg =0;
    while (!average.isEmpty()){
        avg += average.pop();
    }
    System.out.println("The Average pairs for N "+ N + " is "+ avg/5);
}
```

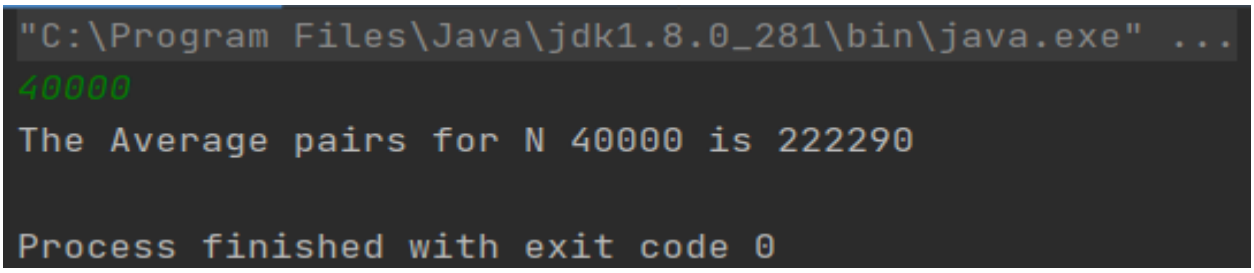
Output:



N = 10000



N = 20000



N = 40000

Part 3

After trying with different large values of N and by doubling hypothesis M is approximately N to the power of 1.18

M ~ N ^ 1.18

We can deduce this to

$$\boldsymbol{M = (N * ln(N))/2}$$