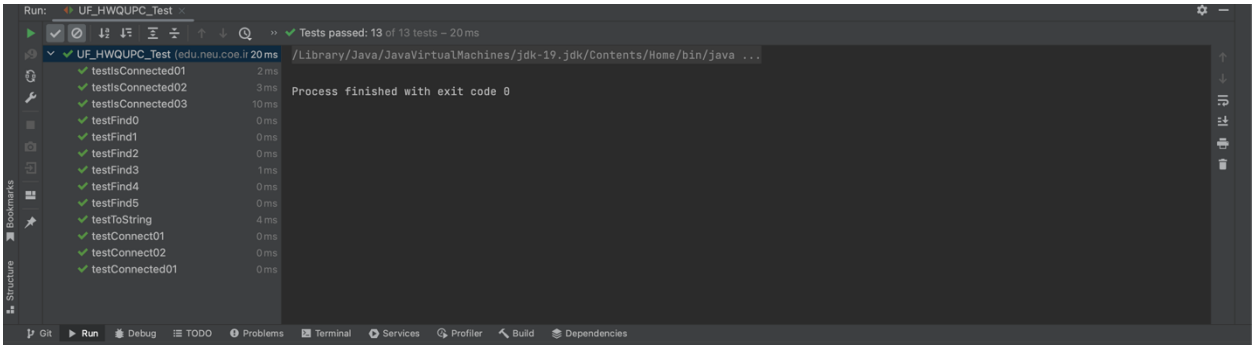
Assignment 4

NUID: 002747795
Name: Keerthana Satheesh

Test Case Screenshot:



Code:

```
no usages  xiaohuanlin *
public int find(int p) {
    validate(p);
    int root = p;
    // FIXME
    while(root != parent[root]){
        if(pathCompression){
            doPathCompression(root);
        }
        root = parent[root];
    }
    // END
    return root;
}
```

```java
private void mergeComponents(int i, int j) {
    // FIXME make shorter root point to taller one
   if(i==j) return;

   if(height[i]<height[j]) {
       updateParent(i,j);
       updateHeight(j,i);
   } else {
       updateParent(j,i);
       updateHeight(i,j);
   }
    // END
}

/**
 * This implements the single-pass path-halving mechanism of path compression
 */
1 usage  xiaohuanlin *
private void doPathCompression(int i) {
    // FIXME update parent to value of grandparent
    updateParent(i, getParent(getParent(i)));
    // END
}
```

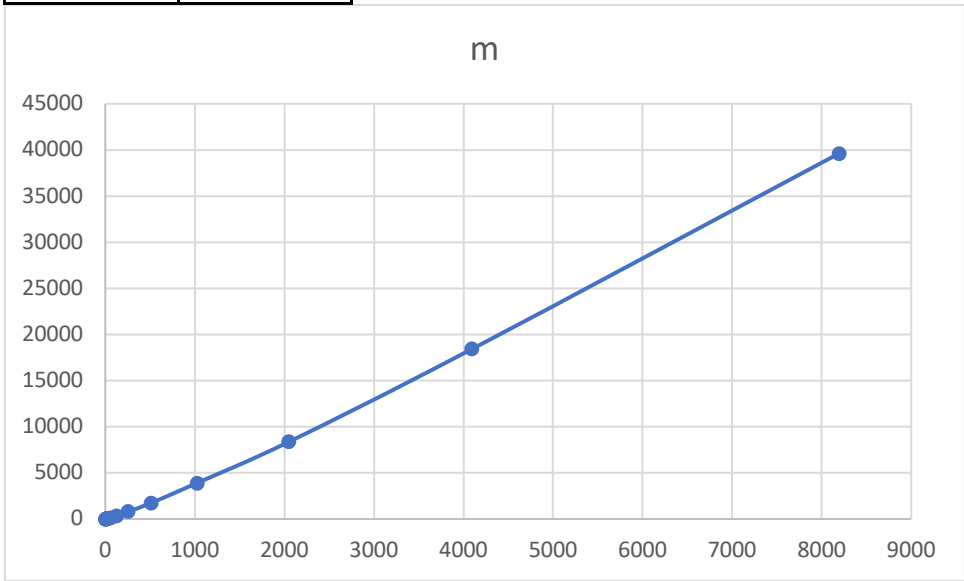
```java
private static int count(int n){
    int count = 0;
    UF_HWQUPC uf = new UF_HWQUPC(n);
    Random random = new Random();
    while(uf.components()!=1){
        int a = random.nextInt(n);
        int b = random.nextInt(n);
        uf.connect(a,b);
        count++;
    }
    return count;
}

no usages  new *
public static void main(String[] args) {
    int times = 100;
    for(int n = 1; n <= 10000; n*=2){
        long sum = 0;
        for(int i = 0; i < times; i++){
            sum += count(n);
        }
        long connections = sum/times;
        System.out.println("N Values: " + n  + " no of connections: " + connections);
    }

}
```

Relationship :

n	m
1	0
2	1
4	5
8	13
16	28
32	67
64	155
128	335
256	801
512	1739
1024	3906
2048	8358
4096	18431
8192	39608



The relationship between the number of objects (n) and number of pairs (m) generated to reduce the number of components from n to 1
$m = f(n) = (n * \ln(n)) / 2$

In union find we check if pairs are connected or disconnected (n (ln(n)),
since there are only 2 possibilities for each pair.
Hence the relationship between m and n is almost identical to $0.5 * (n * \ln(n))$