

Program Structures and Algorithms

Spring 2023(SEC –8)

NAME: Junlong Qiao
NUID: 002784609

Task: Assignment 4(WQUPC)

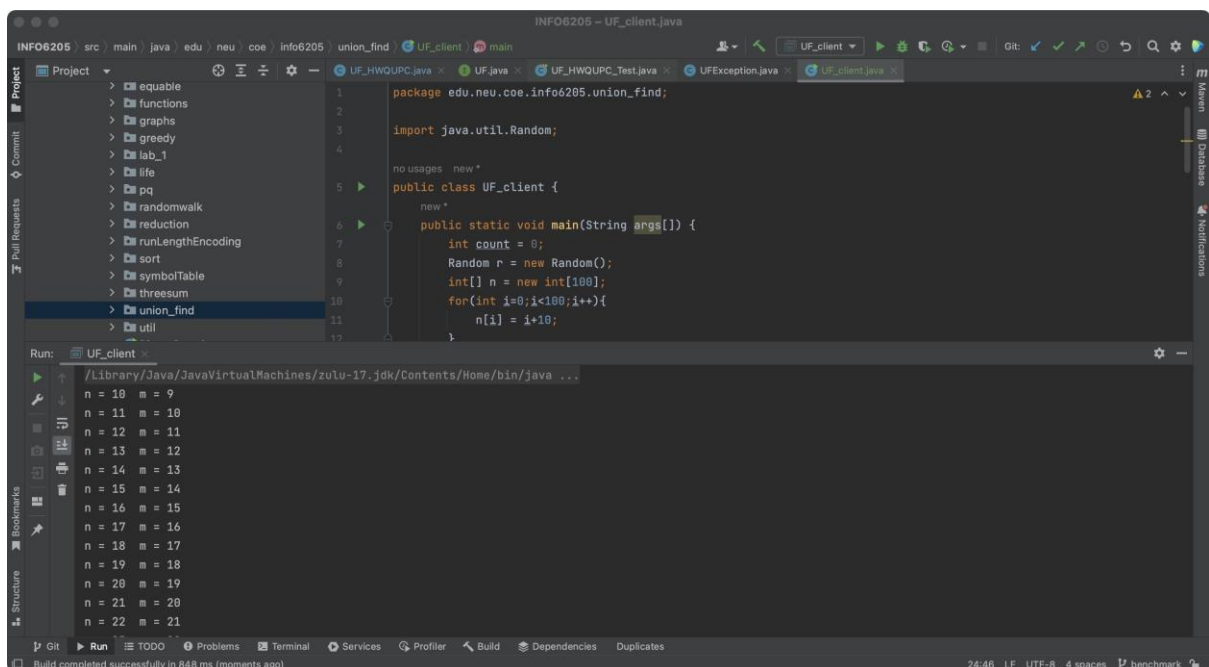
Implement height-weighted Quick Union with Path Compression and check that the unit tests for this class all work. Develop a UF client to find the relationship between the number of objects and the pairs

Runtime Relationship Conclusion:

If all sites in the union are connected, it means the union becomes a connected undirected graph with least edges. Connecting edges of undirected graphs must greater than or equal to its vertices. Therefore, $n = m + 1$.

$$n = m + 1$$

Evidence to support that conclusion:



The screenshot shows an IDE with a project named 'INFO6205'. The project structure includes a 'union_find' package. The 'UF_client.java' file is open, showing the following code:

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

import java.util.Random;

public class UF_client {
    public static void main(String args[]) {
        int count = 0;
        Random r = new Random();
        int[] n = new int[100];
        for(int i=0; i<100; i++){
            n[i] = i+10;
        }
    }
}
```

The output of the program is displayed in the 'Run' console, showing the relationship between the number of objects (n) and the number of pairs (m):

```
n = 10 m = 9
n = 11 m = 10
n = 12 m = 11
n = 13 m = 12
n = 14 m = 13
n = 15 m = 14
n = 16 m = 15
n = 17 m = 16
n = 18 m = 17
n = 19 m = 18
n = 20 m = 19
n = 21 m = 20
n = 22 m = 21
```

The output confirms the relationship $n = m + 1$.

The screenshot shows an IDE window titled 'INFO6205 - UF_client.java'. The code in the editor is as follows:

```

package edu.neu.coe.info6205.union_find;

import java.util.Random;

no usages new *
public class UF_client {
    new *
    public static void main(String args[]) {
        int count = 0;
        Random r = new Random();
        int[] n = new int[100];
        for(int i=0; i<100; i++){
            n[i] = i+10;
        }
    }
}

```

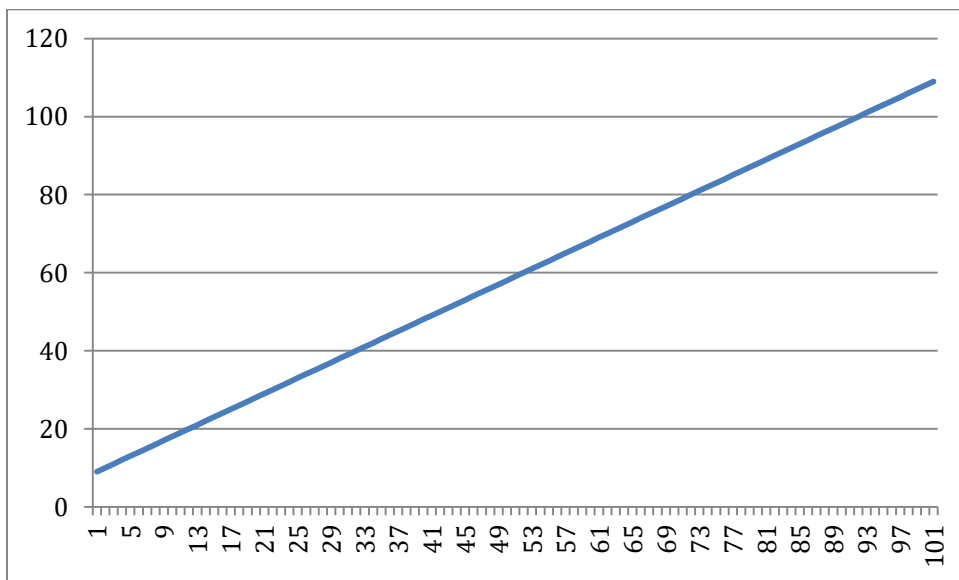
The 'Run' output at the bottom shows the following data:

```

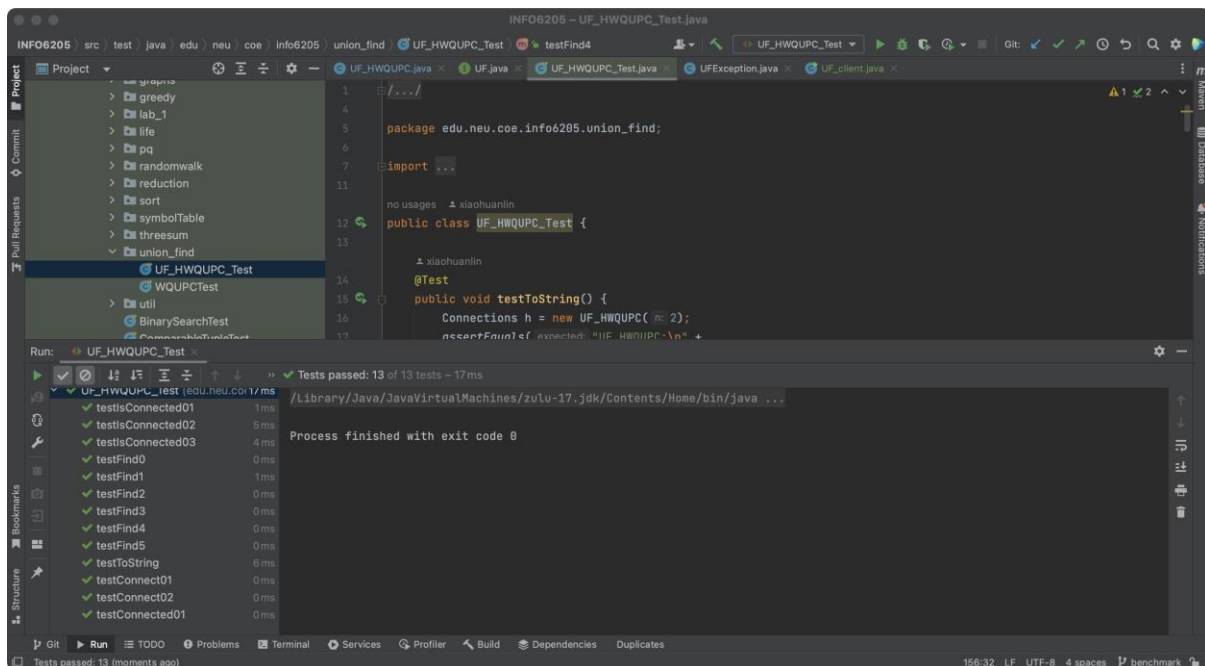
n = 96 m = 95
n = 97 m = 96
n = 98 m = 97
n = 99 m = 98
n = 100 m = 99
n = 101 m = 100
n = 102 m = 101
n = 103 m = 102
n = 104 m = 103
n = 105 m = 104
n = 106 m = 105
n = 107 m = 106
n = 108 m = 107
n = 109 m = 108

```

n	m
	9
	10
	11
	12
	13
	14
	15
	16
	17
	18
	19
	...
110	109



Unit Test Screenshots:



Code of 3-Sum:

UF-client.java

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

import java.util.Random;

public class UF_client {
    public static void main(String args[]) {
        int[] n = new int[100];
        for(int i=0;i<100;i++){
            n[i] = i+10;
        }
        for (int i = 0; i < n.length; i++) {
            int m = count(n[i]);
            System.out.println("n = " + (i+10) + " m = " + m);
        }
    }

    private static int count(int n){
        Random r = new Random();
        UF h = new UF_HWQUPC(n);
        int count = 0;
        while (!isAllConnect(h, n)) {
            int p = r.nextInt(n);
            int q = r.nextInt(n);
            if (!h.isConnected(p, q)) {
                h.union(p, q);
                count++;
            }
        }
    }
}
```

```
    }  
  }  
  return count;  
}  
private static boolean isAllConnect(UF h, int n){  
  for(int i=1;i<n;i++){  
    if(!h.isConnected(0,i)){  
      return false;  
    }  
  }  
  return true;  
}  
}
```