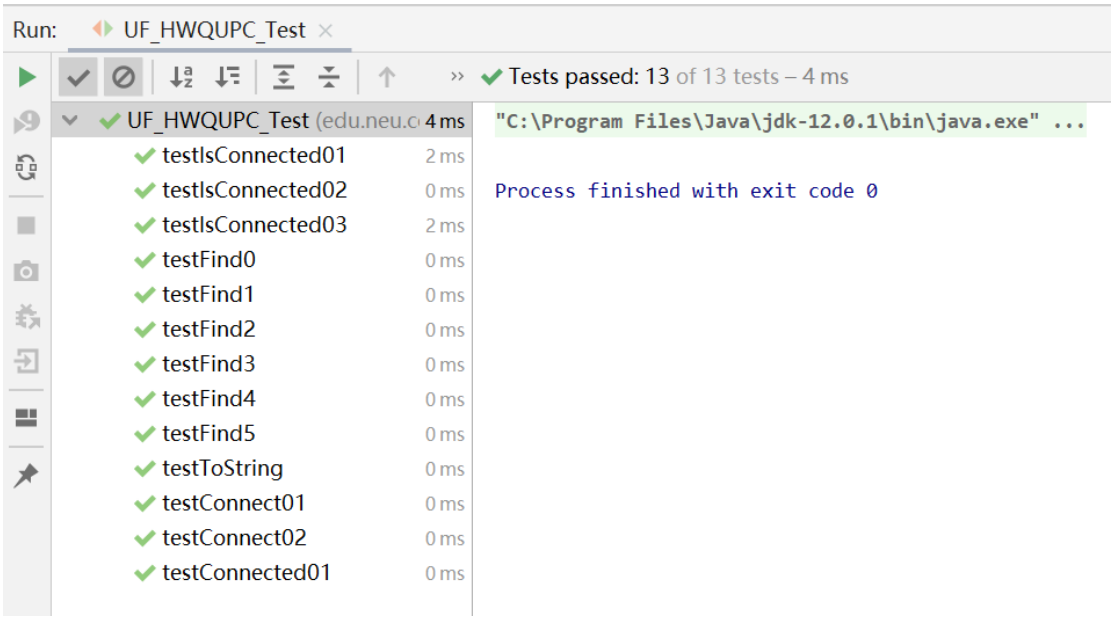
1. UF_HWQUPC_Test screenshot



2. Main() and count() with given n from 10000 to 1, which is also used to determine the relationship between the number of objects (n) and the number of pairs (m) generated.

Results printing out:

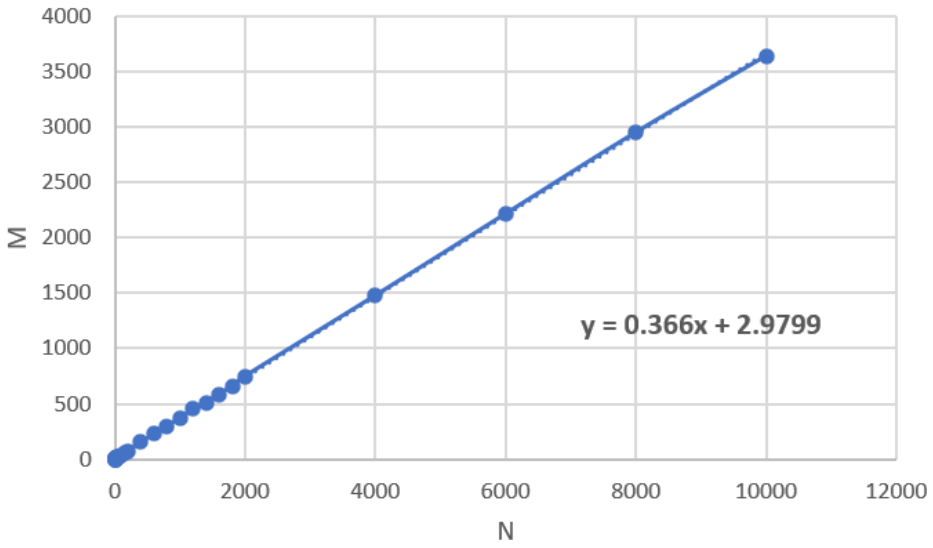
```
N: 10000, number of collection M: 3636
N: 8000, number of collection M: 2947
N: 6000, number of collection M: 2212
N: 4000, number of collection M: 1476
N: 2000, number of collection M: 746

 N: 2000, number of collection M: 749
 N: 1800, number of collection M: 659
 N: 1600, number of collection M: 585
 N: 1400, number of collection M: 512
 N: 1200, number of collection M: 452
 N: 1000, number of collection M: 369
 N: 800, number of collection M: 301
 N: 600, number of collection M: 228
 N: 400, number of collection M: 153
 N: 200, number of collection M: 70
```

```
N: 200, number of collection M: 74
N: 150, number of collection M: 56
N: 100, number of collection M: 36
N: 50, number of collection M: 18
N: 40, number of collection M: 18
N: 30, number of collection M: 10
N: 20, number of collection M: 7
N: 10, number of collection M: 4
N: 9, number of collection M: 6
N: 8, number of collection M: 4
N: 7, number of collection M: 3
N: 6, number of collection M: 2
N: 5, number of collection M: 3
N: 4, number of collection M: 3
N: 3, number of collection M: 1
N: 2, number of collection M: 2
N: 1, number of collection M: 1
```

3. From the data I collected, the graph and fitting line are drawn. After sufficiently large and much sampling experiments on N, I found that number of connections M generated decreases with the decrease of number of objects N, showing a linear relationship. The approximate fitting expression is as follows: $M = 0.366N + 2.979$.

   The performance of weighted quick union improved by path compression is obviously better than the previous methods, and it takes less time. Theoretically speaking, the relationship should based on the iterated log function, but according to our experimental results, it tends to be more linear, which is reasonable and convincing.


4000
3500
3000
2500
2000
1500
1000
500
0
M
0
2000
4000
6000
8000
10000
12000
N
y = 0.366x + 2.9799