Sutiksh Verma

002122052

INFO 6205

Program Structures & Algorithms

Fall 2021

Assignment No. 3

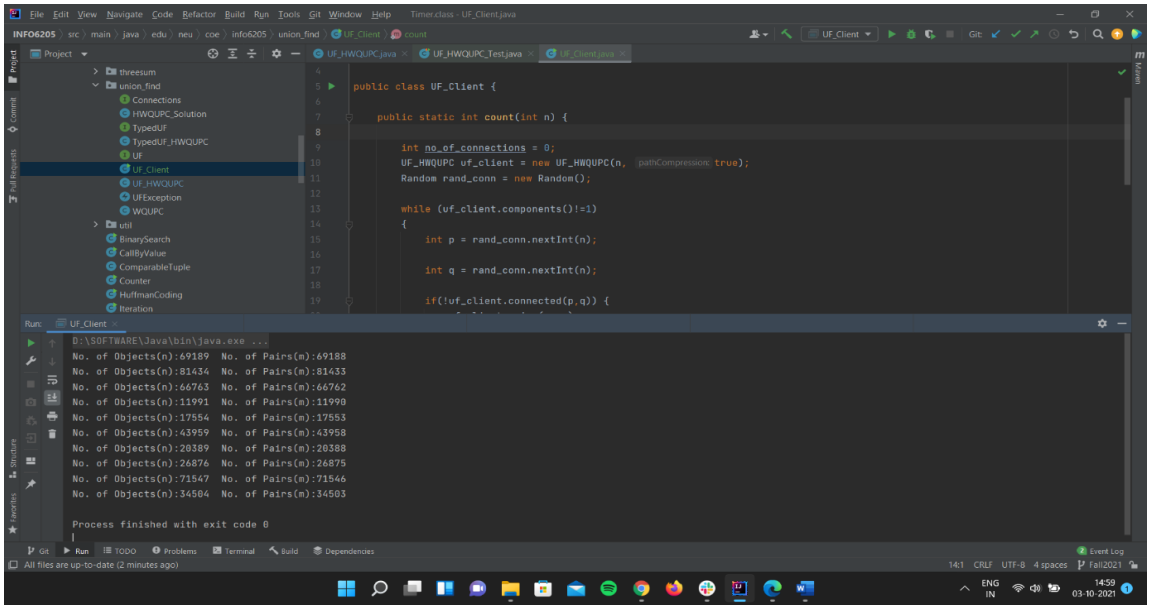
Task: To implement height-weighted Quick union with Path compression.

For this task UF_HWQUPC java class was used and following methods were implemented

- find () – method to update the root of input object if pathCompression is performed
- mergeComponents() – method to merge 2 subtrees such that smaller root points to larger root
- doPathCompression() – method that implements the single-pass process of pathCompression method.

Also, UF_Client java class was created to perform and test the implementation of UF_HWQUPC class

Output:



Console Output:

No. of Objects(n):69189 No. of Pairs(m):69188

No. of Objects(n):81434 No. of Pairs(m):81433

No. of Objects(n):66763 No. of Pairs(m):66762

No. of Objects(n):11991 No. of Pairs(m):11990

No. of Objects(n):17554 No. of Pairs(m):17553

No. of Objects(n):43959 No. of Pairs(m):43958

No. of Objects(n):20389 No. of Pairs(m):20388

No. of Objects(n):26876 No. of Pairs(m):26875

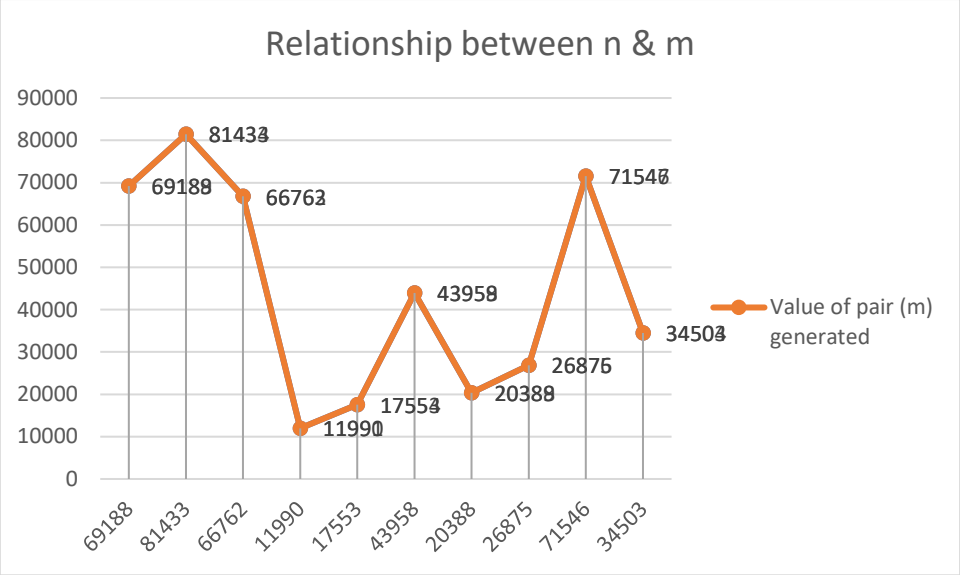
No. of Objects(n):71547 No. of Pairs(m):71546

No. of Objects(n):34504 No. of Pairs(m):34503

Relationship: It can be concluded from the results mentioned above that the number of pairs(m) generated are proportional to the number of objects provided as input. i.e $m = n - 1$

Evidence: I have attached a table and a chart to show the relationship between the value of object (n) and value of pair (m) generated with different set of values for both n and m. As a result, we can see the proportionate result between n and m.

Value of Object (n)	Value of pair (m) generated
69189	69188
81434	81433
66763	66762
11991	11990
17554	17553
43959	43958
20389	20388
26876	26875
71547	71546
34504	34503



The left side of the chart is the value of object (n)

Screenshots of Passed Unit tests: I have attached the screenshot of successfully passed unit test for the class UF_HWQUPC test class.

UF_HWQUPC_Test.java