**Venkata Sai Akhil Vemula**

**002981289**

**INFO 6205 Program Structures and Algorithms**

**Assignment 3**

**Task 1:  Implement height-weighted Quick Union with Path Compression**

Image 1.1: find method implementation in UF\_HWQUPC.java

Text

Description automatically generated

Image 1.2: mergeComponents and doPathCompression ImplementationText

Description automatically generatedImage 1.3: All test cases passedText

Description automatically generated

**Step 2: Using your implementation of UF\_HWQUPC, develop a UF ("union-find") client**

Created class named UnionFindClient in the same package as UF\_HWQUPC and implemented count and main static methods. Took initial n= as 17 and incremented every time by 17 for 30 times. Made 10 experiments/loops for generating average no of pairs that are required to complete Union.

Image 2.1 UnionFindClient class count ImplementationText

Description automatically generated

Image 2.2: UnionFindClient class Main Method implementation

Text

Description automatically generated

Image 2.3: Output of runs

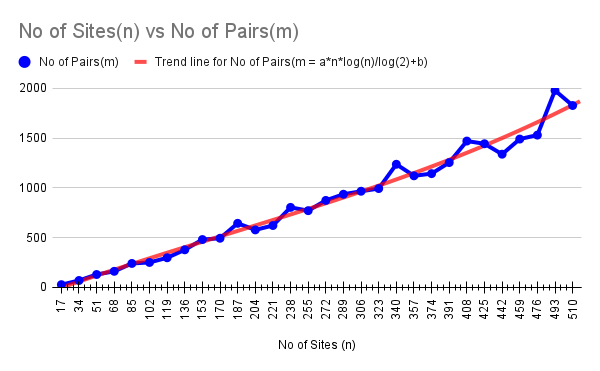
Text

Description automatically generated

**STEP 3 IN NEXT PAGE**

**Step 3: (Evidences & Conclusion)**Determine the relationship between the number of objects (*n*) and the number of pairs (*m*)

Graph 1: n vs m

**.**

Google Sheet Link <https://docs.google.com/spreadsheets/d/1ed8vp7l4cgqy1cxlZZ5VHia-kUyPI-xsJiSpBcisW40/edit?usp=sharing>

\*\* CSV file is in the same directory as this report

**Conclusion**: It is observed that the values(no of pairs) obtained from the experiment are almost equal to NLog2(N) in most of the cases. The same trend was observed in the trend line which is generated by plotting a graph in google sheet(Trend line was plotted using google sheet in-built functions). Though total no of pairs are almost equal to above said values there is some fluctuation due to the randomness. Hence it can be concluded that

**M = A\*N\*Log2(N) + B**

Where

M = total number of pairs

N = total no of sites/nodes

A = co-efficient (According to google sheet this is equal to 0.65433, however this is changing in every main method run)

B = co-efficient