For Alternative 1:

We store the depth rather than the size. Looking at the constructor of this class, initializing this class takes O(N) times. Then for each method, Union, Find and Connected take log(N) times, which is a great improvement of Quick find.

For Alternative 2:

Since whenever we call Find(n) method, we have to traverse the path from n to root. So, along the way we can connected all the items we visited to their root. After implementing the algorithm, we analyze the running time of each method. Initializing the algorithm takes O(N) times. Then for each method, Union, Find and Connected take log\* n times. log\* n is the iterated log function, which is significant improvement. Even thought when N is 2^65536,

log\* n is 5. So, we can say that for any real-word input, log\* n is less than 5. So, for each method, Union, Find and Connected, it seems like they almost take constant times. **Compared with the Alternative1, the running time of the methods (Union, Find, and Connected) decrease from logarithmic time to linear time, which is a significant improvement.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm | Initialize | Union | Find | Connected |
| Quick Find | N | N | 1 | 1 |
| Weight QU | N | log n | log n | log n |
| Weight QU Path Compression | N | log\* n | log\* n | log\* n |

WQU\_Alternative1 Test:

Graphical user interface, text, application, chat or text message

Description automatically generated

WQU\_Alternative2 Test:

Graphical user interface, text

Description automatically generated