SAKSHI SHAH (001003644)

**Program Structures & Algorithms**

**Spring 2021**

**Assignment No. 3**

* **Task :**

1. For weighted quick union, store the depth rather than the size;
2. For weighted quick union with path compression, do two loops, so that all intermediate nodes point to the root, not just the alternates.

* **Output:**

**-- PART A**

**1. By height**

-- When we merge two trees with different height, the shorter tree will get connected to the larger one. But when we connect two trees with same height, the height of the parent increases by 1. This means that, in order to increase the height of the tree, the number of nodes has to at least double.

Example,

* For h=1, n=1
* For h=2, n>=2
* For h=3, n>=4
* For h=4, n>=8

Therefore,

**h<=lg(n)**

**2. By size**

-- When we merge two trees by size, the tree with less number of nodes gets connected to the one with greater number of nodes. By doing so, the size of the parent is increases by the size of the child. So, we can say that the number of nodes of the tree is always greater than or equal to 2 raised to its height.

n>=2^h

i . e,

**h<=lg(n)**

That is, no matter weighted by height or by size, their depths have the same upper bound. Therefore, the time taken will be same for both the methods. Hence, there is no need of benchmarking.

* **Relationship Conclusion:**

-- Both the graphs have almost similar values and weighted UF performs a bit better than alternate parent path compression.

* **Graphical representation:**

Chart, line chart

Description automatically generated

* **Unit tests result:**

--Attached below is a screenshot of all the unit test case running successfully.

For

UF\_HWQUPC\_Test.java and WQUPCTest.java

A screenshot of a computer

Description automatically generatedText

Description automatically generated

A picture containing text, screenshot, monitor

Description automatically generatedA picture containing text, monitor, screenshot

Description automatically generated