BLG 223 HW5 -- Ali Emre Kaya -- 150210097

group1 - opADD

As can be clearly seen, the time complexity of skiplist is much higher than other data structures.

group1 - op UPDATE

group1 - op DELETE

group2 - opADD

time(s) vs **employee(k)**

Legend: ARRAY · ARRAY = 7.751E… · LINKE… · LINKE… · VECTOR · VECTOR = 6.091E… · LIST · LIST = 6.234E…

group2 - opUPDATE

time(s) vs **employee(k)**

Legend: ARRAY · ARRAY = 6.407E… · LINKE… · LINKE… · VECTOR · VECTOR = -8.756… · LIST · LIST = -0.036e…

group2 - opDELETE

time(s) vs **employee(k)**

Legend: ARRAY · ARRAY = 7.666E… · LINKE… · LINKE… · VECTOR · VECTOR = -8.109… · LIST · LIST = -0.032e…
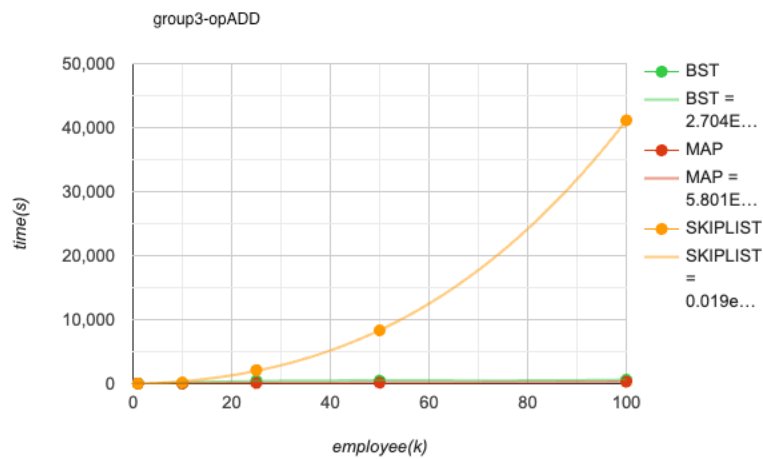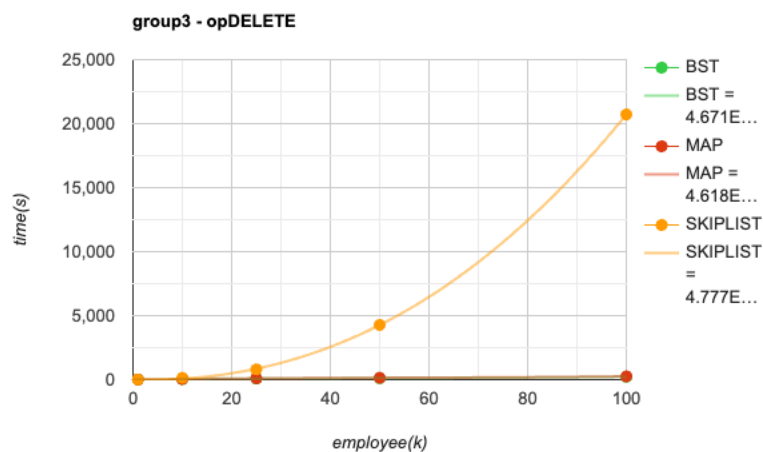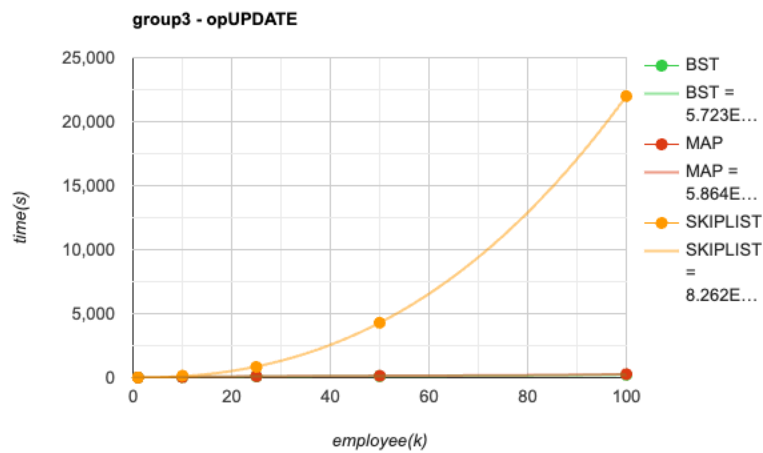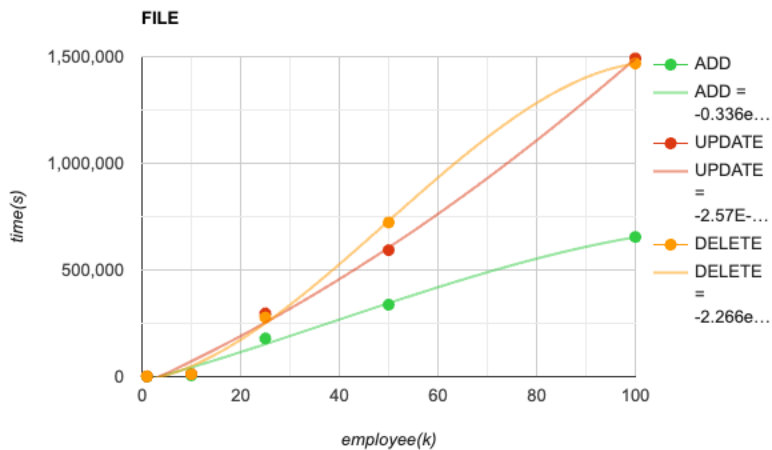
Array, linkedlist, vector, and list's time complexities are almost same for addition, but for update and deletion, list structure remained at a disadvantage and array is the best one in these four.

group3-opADD

●	BST
—	BST = 2.704E…
●	MAP
—	MAP = 5.801E…
●	SKIPLIST
—	SKIPLIST = 0.019e…
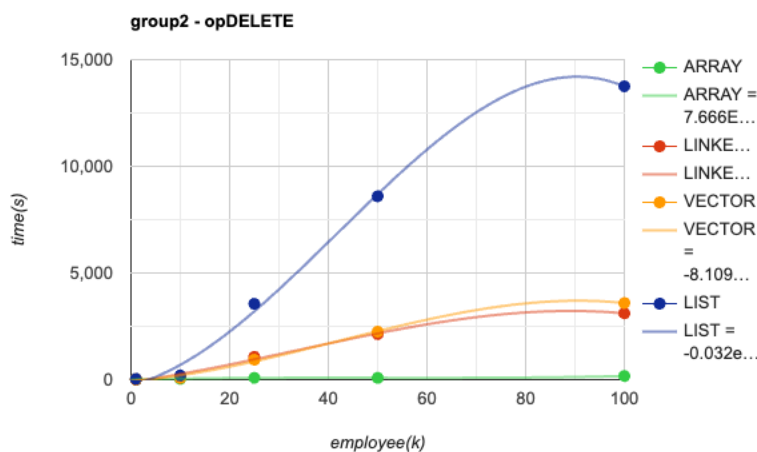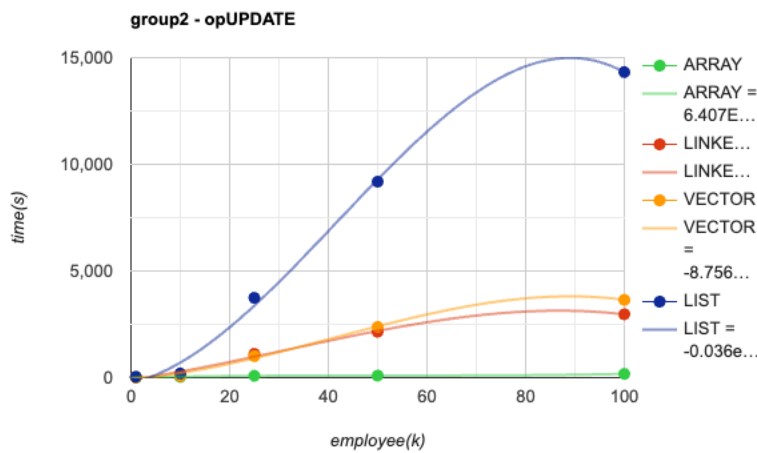
Skiplist again has the largest time complexity, so much so that analyzing other data structures is very difficult in this graph.

group3 - opUPDATE

●	BST
—	BST = 5.723E…
●	MAP
—	MAP = 5.864E…
●	SKIPLIST
—	SKIPLIST = 8.262E…

group3 - opDELETE

●	BST
—	BST = 4.671E…
●	MAP
—	MAP = 4.618E…
●	SKIPLIST
—	SKIPLIST = 4.777E…

FILE

For this assignment, I created dataset and operator creators. I used the dataset and operations I obtained with these to create graphs. I used 100k operations to test the time of files with 50k and 100k employees. I used 10k operations for 25k and 1k operations for lower ones. Since the output of my file code took a very long time and completely corrupted the other graphics, I decided to show it in a separate graphic. The codes and data I use are available in the zip file.

I used this site to plot the data:: https://www.rapidtables.com/tools/scatter-plot.html