A1_P2

By: Ethan Bohling

Task 4:

Splay Tree Variants	Access Pattern			
	Sequential	**Uniform**	**Skewed**	**Working Set**
Bottom-up	1,595,883 rotations Average search depth: 5 0.23 seconds	8,692,081 rotations Average search depth: 20 0.33 seconds	7,792,725 rotations Average search depth: 19 0.31 seconds	2,005,995 rotations Average search depth: 5 0.24 seconds
Top-Down	821,835 rotations Average search depth: 4	5,868,173 rotations Average search depth: 21	1,207,380 rotations Average search depth: 4	5,224,237 rotations Average search depth: 19
Semi-splay (option 1 or 2)	0.21 seconds 1,273,121 rotations Average search depth: 2944 1.27 seconds	0.33 seconds 7,610,523 rotations Average search depth: 20 0.33 seconds	0.22 seconds 7,009,059 rotations Average search depth: 18 0.31 seconds	0.31 seconds 1,785,587 rotations Average search depth: 21 0.22 seconds
Weighted-splay (option 1 or 2)	1,595,883 rotations Average search depth: 5 0.23 seconds	6,793,682 rotations Average search depth: 20 0.34 seconds	5,658,889 rotations Average search depth: 20 0.39 seconds	2,077,086 rotations Average search depth: 15 0.25 seconds

Task 5:

- There is no difference in asymptotic complexity. Both the top-down and bottom-up approaches have the same amortized analysis of Big O(log n) for search and deletion.
- Splay trees are not thread-safe because the search function not only traverses the tree, but it also rearranges the tree. This can create pointer corruption and data corruption.
- The semi-splay operation reduces the number of rotation overhead because it refuses to splay nodes past a certain depth limit. The advantages of a weighted-splay operation are that it protects heavily weighted data from being at the bottom of a tree. Heavily weighted data will be near the top of the tree.
- Task 3 of a weighted-splay turns a standard binary search tree into a self-optimizing frequency cache. By ensuring weight >= parent, the tree decides which data are worth the computational cost of rotation, making it a powerful data structure for real-world databases.