Splay Tree Variants	Sequential	Uniform	Skewed	Working Set
Bottom-Up	541425 Rotations Depth: 4.41426 35486 microseconds	5784242 rotations Depth: 23.5972 323553 microseconds	4724391 rotations Depth: 19.5238 251470 microseconds	1062668 rotations Depth: 5.25846 40871 microseconds
Top-Down	531961 rotations Depth: 5.31961 12276 microseconds	1220714 rotations Depth: 13.9832 22009 microseconds	1219178 rotations Depth: 12.8811 20411 microseconds	743427 rotations Depth: 10.7336 14565 microseconds
SemiSplay (option 1 rotation limit of 3)	199971 rotations Depth: 9088.32 12261 microseconds	1181361 rotations Depth: 19.0032 21718 microseconds	1174595 rotations Depth: 16.0133 20518 microseconds	729800 rotations Depth: 70.8313 16508 microseconds
Weighted-Splay (option 2)	531962 rotations Depth: 5.31961 22009 microseconds	1225307 rotations Depth: 13.5431 22155 microseconds	1225766 rotations Depth: 12.998 21667 microseconds	723412 rotations 10.6186 15228 microseconds

Task 5:

1. Both the top-down and bottom-up approaches have the same amortized cost on paper of log(n) so they should not have any difference but practically they will behave differently due to the bottom-up approach requiring parent nodes and the top-down approach doing the opposite resulting in a slight difference in the data sets. Top-down will also save more memory.
2. Splay trees are not thread safe as every single operation we have for the trees will alter the structure of the tree in some way through our rotations meaning that if multiple functions were accessed at the same time, it could really mess up the tree.
3. SemiSplay helps us in cases where the data is random as limiting the rotations will allow for less overhead cost as we will not be splaying up to the root every time. So in larger

randomized data sets, semi splay trees are incredibly useful. Weighted splay trees allow us to keep data that is accessed more frequently near the top so that way it is easier to reach in future passes, hence the weight of the node.

4. For the sequential access pattern, the full splay trees have trouble with how many times the tree is rotating since every operation splays the node that was accessed up to the root of the tree. The semi splay trees struggle with the average depth since the limit (in this case 3) doesn't allow for a full rebalancing of the tree in some cases. The weighted splay trees don't help for this one since each node will only be accessed a single time. Uniform makes the full splay approach moderately successful since the random access makes the tree have a relative balance from the splays. This is where the semi-splay excels since the limited rotations will be cut down on time without making the average depth go incredibly high. Weighted splays don't really provide an advantage for this access pattern since all nodes are accessed equally. For the skewed access pattern, the full splay operations still generate a lot of unnecessary time rotating nodes that are less accessed. Semi-splay can struggle here as well since not all nodes will be near the root due to the limited amount of splaying. This is where weighted splaying is best as the weighted splay allows for the nodes that have high weights and are accessed a lot to stay near the top of the tree. Working set has good performance in the full splay trees since all nodes stay near the root after being splayed the first time. Semi-Splay has ok performance due to the limit on the rotations. Weighted splay does well here for the same reason as lest time, the weighted nodes will stay near the root allowing for easier access.