[Avg inserted]

randomized	type	count	
False	AVL	10000	9872
		15000	14723
		20000	19486
		25000	24207
		30000	28836
	BST	10000	9872
		15000	14723
		20000	19486
		25000	24207
		30000	28836
	LIST	10000	9871
		15000	14710
		20000	19484
		25000	24194
		30000	28809
True	AVL	10000	10000
		15000	15000
		20000	20000
		25000	25000
		30000	30000
	BST	10000	10000
		15000	15000
		20000	20000
		25000	25000
		30000	30000
	LIST	10000	10000
		15000	15000
		20000	20000
		25000	25000
		30000	30000

[Avg insert time]

randomized	type	count	
False	AVL	10000	578.46
		15000	1484.96
		20000	2854.10
		25000	4775.43
		30000	7216.46
	BST	10000	4.13
		15000	6.07
		20000	7.72
		25000	9.41
		30000	11.09
	LIST	10000	522.20
		15000	1279.36
		20000	2529.37
		25000	4160.68
		30000	6249.87
True	AVL	10000	1418.78
		15000	3586.11
		20000	6842.56
		25000	11181.17
		30000	15772.24
	BST	10000	3.30
		15000	5.73
		20000	8.10
		25000	10.57
		30000	13.57
	LIST	10000	923.75
		15000	2492.36
		20000	4935.27
		25000	8232.16
		30000	12337.00

[Avg found]

randomized	type	count	
False	AVL	10000	1041
		15000	2203
		20000	3885
		25000	6030
		30000	8616
	BST	10000	1041
		15000	2203
		20000	3885
		25000	6030
		30000	8616
	LIST	10000	957
		15000	2131
		20000	3844
		25000	6029
		30000	8603
True	AVL	10000	998
		15000	2252
		20000	3913
		25000	6209
		30000	8930
	BST	10000	998
		15000	2252
		20000	3913
		25000	6209
		30000	8930
	LIST	10000	1007
		15000	2236
		20000	4050
		25000	6265
		30000	9081

[Avg search time]

randomized	type	count	
False	AVL	10000	1.11
		15000	2.23
		20000	3.37
		25000	4.80
		30000	6.06
	BST	10000	1.94
		15000	3.19
		20000	4.34
		25000	5.74
		30000	7.36
	LIST	10000	850.96
		15000	1988.71
		20000	3779.22
		25000	5895.82
		30000	8703.29
True	AVL	10000	1.29
		15000	2.64
		20000	3.89
		25000	5.40
		30000	6.64
	BST	10000	1.23
		15000	3.00
		20000	4.17
		25000	5.64
		30000	6.95
	LIST	10000	896.67
		15000	2189.11
		20000	4055.83
		25000	6583.80
		30000	9657.00

[Avg removed]

randomized	type	count	
False	AVL	10000	914
		15000	2006
		20000	3520
		25000	5340
		30000	7478
	BST	10000	914
		15000	1999
		20000	3425
		25000	5228
		30000	7288
	LIST	10000	879
		15000	2022
		20000	3474
		25000	5392
		30000	7538
True	AVL	10000	965
		15000	2065
		20000	3636
		25000	5469
		30000	7819
	BST	10000	965
		15000	1974
		20000	3528
		25000	5469
		30000	7819
	LIST	10000	936
		15000	2111
		20000	3629
		25000	5588
		30000	7725

[Avg remove time]

randomized	type	count	
False	AVL	10000	371.99
		15000	1353.03
		20000	3199.50
		25000	5936.05
		30000	9840.83
	BST	10000	3.47
		15000	6.12
		20000	8.47
		25000	10.85
		30000	13.61
	LIST	10000	1315.24
		15000	3039.27
		20000	5444.87
		25000	8361.52
		30000	11737.62
True	AVL	10000	480.44
		15000	1653.11
		20000	4001.87
		25000	7470.06
		30000	12241.17
	BST	10000	3.36
		15000	5.78
		20000	8.04
		25000	11.07
		30000	13.72
	LIST	10000	1710.51
		15000	3943.24
		20000	7043.89
		25000	10689.59
		30000	15332.95

Results summary:
	- For inserting, The most effective is BST tree (400x faster than second LIST)
	- For inserting, The least effective is AVL tree (probably due to tree rotations
	- For searching, The most effective is AVL tree
	- For searching, The least effective is LIST (1000x slower than second BST)
	- For removing, The most effective is BST tree (500x faster than second AVL)
	- For removing, The leat effective is LIST

Conclusions:
	- If we want to keep overall performance on high level, we should use BST tree
	- If we want to focus mainly on searching in tree (without mutating it), we should use AVL tree
	- If we don't care about performance but we focus on algorighm simplicity, we should use LIST