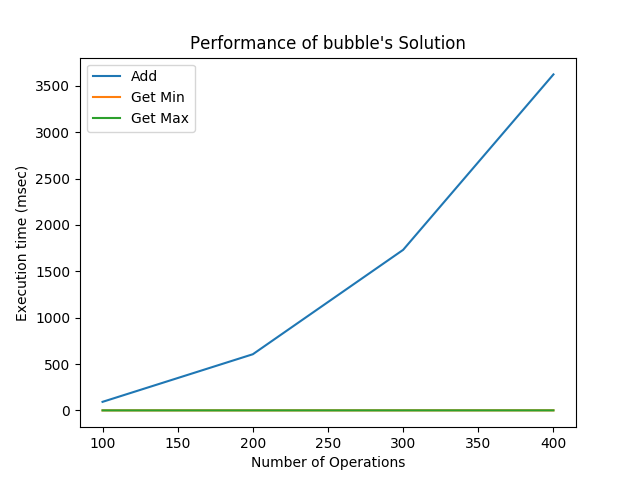
SORTING IN DIFFERENT WAYS

In Python there are some ways to **Sort** a given list of numbers. I’ll analize 4 different methods: **‘BubbleSort’,‘QuickSort’,’Heap’,’BinaryTree’.** Each of them is able to sort a list of items and also to find the **Max** (maximun value) and **Min** (minum value).

**‘BubbleSort’**:

It goes through the list, compares adjacent pairs and switch them if they are in the wrong order.

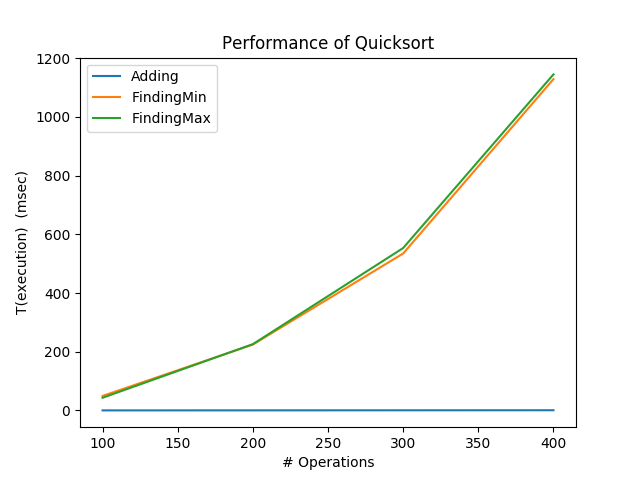


BubbleSort has a **worst case** of O(n2).

When the list is already sorted, the Best Case of bubble sort is only *O*(n). As we can understand from the graph: ‘Add’ takes a lot of time, while ‘Get Min’ and ‘Get Max’ are actually really fast.

**QuickSort**:

It picks an element, called a ***pivot****.* Then It is able to devide the list of numbers by placing bigger numbers on the right, and smaller ones on the left of the pivot. These 2 simple steps are repeated until the list is completely ordered.



As we can see in the graph, Adding is really fast, while finding the Max and the Min takes a little more time

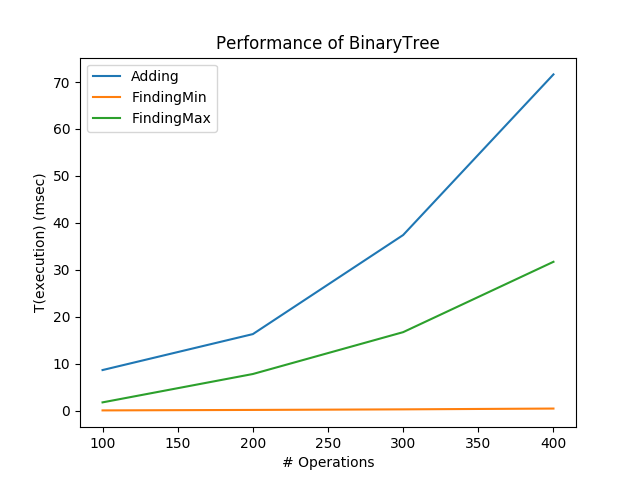
The **best case** is the one when we make only log2 *n* calls before we reach a list of size 1.

On **avarage** It takes *O*(*n* log *n*) time in expectation.

While in the **worst case** it makes O(*n*2) comparisons.

Binary Tree:

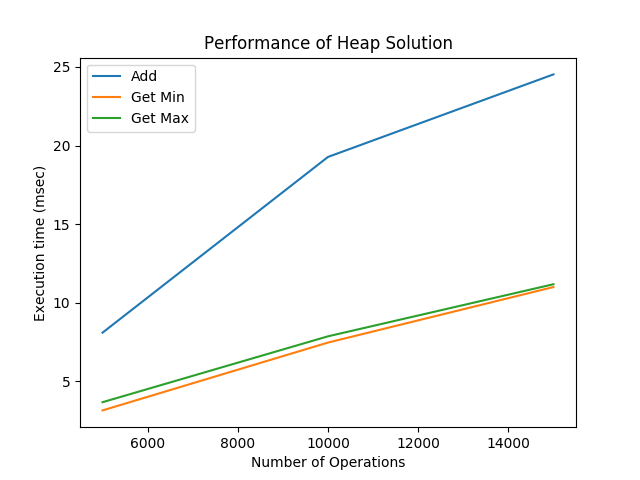
In computer science, a **binary tree** is a **TREE** data structure in which each node has at most two children, which are referred to as the left child and the right child.



On **average**, binary search trees with n nodes have O(logn) height. However, in the worst case, binary search trees can have O(n) height. Adding depends on the number of nodes, while as it can be seen, the Mi realy easy to be found in any case.

**Heap**:

It  Tree based data structure; this method is able to organize a list of numbers by creating **nodes**: the parent node can only be greater than the children nodes. In this way we can easly find the Max and Min values.



As we can see from the graph, the functions to find the Max and Min are really fast compared to the contrast situation of the Add value. The number of operations required depends only on the number of levels the new element must rise to satisfy the heap property, thus the insertion operation has a **worst case**time complexity of O(log n) but an **average-case** complexity of O(1).