Homework 7 by Timofei Podlorytov

7.1

(a-c)
The implementations are provided in the quicksort.cpp file.
The output of the code is:

```
Average time of Lomoto partition: 5.69557e-05
Average time of Median partition: 6.69592e-05
Average time of Hoare partition: 5.17366e-05
```

d)
As we see Hoare performs the best. However, despite the expectation that median will outpace the Lomto we didn't get the difference we wanted. I think that despite better tree height the additional operations cause the overall performance to be worse.

The unexpected difference in relationship between Lomoto and Median Partition can be explained by the search for the Median number taking time and thus adding to the time and even though with worst case input for lomoto it takes less time, but with random sequences its actually worse since we take time to generate the pivot.

7.2

(a) and (c)- the implementations are in the cpp file.

```
Average time of First 2 partition: 0.000331488
Average time of Random 2 partition: 0.000351191
```

b) $T(n)=3T(n/3)+\Theta(n)$ the best case and the partitioning is almost equal in size.
we can use the master method:
$\log_3 3=1$. Then it's the second case $n=n^1$ and $T(n)=\Theta(n*\log_3(n))$-means the performance is nlog(n).-best
In the worst case the algorithm is reversed and thus there's only one partition every single time.
$T(n)=1(T(n-2))+\Theta(n)=\Theta(n)+\Theta(n-2)+...+\Theta(2)=\Theta(n*(n/2))=\Theta(n^2)$-worst case performance.

I think due to the search of the random indices in the array the execution time of the second implementation is actually longer than the first.