Linear Select program analysis:-

- Mean calculating recursive call ~~no~~ does not exceed worst case running time.
- Only 1/5th of the integers provided, are used while other recursive calls uses at most 7/10th the integers provided.

$$\Rightarrow T(n) \leq \underbrace{T(n \times 2/10)}_{\text{Finding true median}} + \underbrace{T(n \times 7/10)}_{\text{Quickselect recursions}} + \underbrace{c \cdot n}_{\text{partitioning work}}.$$

$$\Rightarrow T(n) \leq 10 \cdot c \cdot n \in O(n).$$

Personally, my LinearSelect program uses the Basic skeleton structure from the QuickSelect program provided, so a lot of the code is similar, however the key diference lies in it's choice of pivots. While QuickSelect uses a randomizer to pick a pivot from the integers (50% chance of getting a good pivot), LinearSelect uses the median of Medians algorithm.

As shown above, the median of Medians algorithm picks only 20% of the data and recurses on it till it finds the best median in the whole set of numbers. Linear Select calls other methods more often because data and functions are needed to partition, whereas QuickSelect picks a random pivot with just 2 lines of cooling. In theory, Linear Select is faster, however, while running the 2 programs, QuickSelect is the clear winner.

PS:- Math.floor() & Math.ceil tends to round off more than we need them to :(