Braeden Nelson
ID# 11685416

350 HW 3

(1)
```
min = 1000; max = -1000;
from i=0 to n
    if a[i] > max
        max = a[i]
    if a[i] < min
        min = a[i]
done
```

(2) LinearSelect = $O(i \cdot n)$ avg run time // $S(A,n,i)$
MergeSort = $O(n \log n) + O(1)$ (constant) avg run time // $T(A,n,i)$

Algorithm S performs much better than Algorithm T for values when $i << n$. Because Alg. S depends on i and n, while Alg. T depends only on size of input, n, when $i << n$ Alg. S will be the better choice.
Algorithm T performs better than Algorithm S for values where i is close to the size of n. Once i becomes too large, Algorithm S will begin to perform in roughly $O(n^2)$ while Algorithm T will remain $O(n \log n)$.

(3) Complexity analysis for Worst-case linear Select if $K=3$.
Each group has a median, & we have at least 2 numbers in each group less or equal to median. totally we have $n/3$ medians and among them we have $n/10$ medians less or equal to MM. So in original A[1..n], we have atleast $2 \cdot \frac{n}{10}$ numbers less or equal to MM.
Since every # in low is $\leq MM$, $|low| \geq \frac{2n}{10}$
And $|low| + |high| \leq n$.
Therefore $|low| \leq \frac{8n}{10} \geq |high|$ and $\max\{|low|, |high|\} \leq \frac{8n}{10}$.
Step (1) write a formula: $T_w(n) = T_w\left(\frac{n}{3}\right) + T_w\left(\frac{8n}{10}\right) + O(n)$
Step (2) Guess: $T_w(n) = O(n)$, That is, $\exists c > 0, T_w(n) \leq c \cdot n$ for all n.
I.H: $\forall i < n, T_w(i) \leq c \cdot i$

~~Step 3 on bulk:~~

Step 3 on bulk:

step ③ : check:

$$T_w(n) = T_w\left(\frac{n}{3}\right) + T_w\left(\frac{3n}{10}\right) + a \cdot n$$
$$\leq c \cdot \frac{n}{3} + c \cdot \frac{8n}{10} + a \cdot n$$
$$\leq c \cdot \frac{10n}{30} + c \cdot \frac{24n}{30} + a \cdot n$$
$$\leq \frac{34}{30} \cdot cn + a \cdot n$$
$$\leq c \cdot n \text{ when } c >> a$$

Therefore, Worst-case time complexity is $O(n)$

(complexity analysis for worst case linear select if $k=7$.

Each group has a median & at least 4 numbers in each group less or equal to the median. Totally we have $\frac{n}{7}$ medians & among them, $\frac{n}{10}$ medians less or equal to MM. For each median, we have 4 numbers in the group the median belongs to that are less or equal to median. So, in original $A[1...n]$ we have at least $4 \cdot \frac{n}{10}$ numbers less or equal to the MM.

- Since every # in low is $\leq$ MM. $|low| \geq \frac{4n}{10}$.
- And $|low| + |high| \leq n$.

Therefore $|low| \leq \frac{6n}{10} \geq |high|$ and $\max\{|low|, |high|\} \leq \frac{6n}{10}$.

step ① write a formula: $T_w(n) = T_w\left(\frac{n}{7}\right) + T_w\left(\frac{6n}{10}\right) + O(n)$

step ② Guess: $T_w(n) = O(n)$. That is, $\exists c > 0$, $T_w(n) \leq c \cdot n$ for all n.

I.H: $\forall i < n$, $T_w(i) \leq c \cdot i$

step ③ Check:

$$T_w(n) = T_w\left(\frac{n}{7}\right) + T_w\left(\frac{6n}{10}\right) + a \cdot n$$
$$\leq c \cdot \frac{n}{7} + c \cdot \frac{6n}{10} + a \cdot n$$
$$\leq c \cdot \frac{10n}{70} + \frac{42n}{70} \cdot c + a \cdot n$$
$$\leq cn \cdot \frac{52}{70} + a \cdot n$$
$$\leq c \cdot n \text{ when } c >> a.$$

Therefore, worst-case time complexity is $O(n)$.

(4) iiSelect (A, n, i)

r = partition (A, 1, n) // $O(n)$

(A) if i == r, ret A[r] // $O(1)$

(B) if i < r, ret quickselect (A, 1, r-1, i) // $O(n^2)_{worst} + O(n)_{avg}$

(C) if i > r, ret linearSelect (A, r+1, n, i-r) // $O(n)$

probabilities

$A = \frac{1}{n}$, $B = \frac{r-1}{n}$, $C = \frac{n-r}{n}$

Worst-Case Time Complexity

Step 1 Formula: $T_w(n) = \max_{1 \le r \le n} \{ O(n) + O(1) + T_w((r-1)^2) + T_w(n-r) \}$

Step 2 Guess: $T_w(n) = O(n^2)$. That is, $\exists c > 0$, such that $T_w(n) \le c \cdot n$ for all n.

I.H: $\forall i < n$, $T_w(i) \le c \cdot i^2$

Step 3 Check:

$T_w(n) = \max_{1 \le r \le n} \{ O(n) + T_w(r-1)^2 + T_w(n-r) + a \cdot n \}$

$\le \max_{1 \le r \le n} \{ c \cdot (r-1)^2 + c \cdot (n-r) + a \cdot n \}$

$F(r) = c \cdot (r-1)^2 + c \cdot (n-r) + a \cdot n$

$F'(r) = 2cr - 3c = 0$, $2r - 3 = 0$, $r = \frac{3}{2}$

$F(1) = cn + an$

$F(n) = c \cdot (n-1)^2 + a \cdot n$

$F(\frac{3}{2}) = c \cdot \frac{1}{4} + c \cdot (n - \frac{3}{2}) + a \cdot n$

$= F(n) = c(n-1)^2 + an = cn^2 - 2cn + 1 + a \cdot n$

$\le cn^2$ when $c >> a$.

Clearly, iSelect has a worst-case time complexity of $O(n^2)$.

Avg-Case Time Complexity

Step 1: Formula: $T_{avg}(n) = \frac{1}{n} \sum_{r=1}^{n} \{ \frac{1}{n} \cdot O(1) + \frac{r-1}{n} \cdot T_{avg}(r-1) + \frac{n-r}{n} \cdot T_{avg}(n-r) + O(n) \}$

Step 2: Guess: $T_{avg}(n) = O(n)$. That is, $\exists c > 0$, such that $T_{avg}(n) \le c \cdot n$ for all n.

I.H: $\forall i < n$, $T_{Avg}(n) \le c \cdot i$

Step 3: Check: $T_{avg}(n) = \frac{1}{n} \sum_{r=1}^{n} \{ \frac{1}{n} \cdot O(1) + \frac{r-1}{n} \cdot T_{avg}(r-1) + \frac{n-r}{n} \cdot T_{avg}(n-r) + O(n) \}$

$\le \frac{1}{n} \sum_{r=1}^{n} \{ \frac{1}{n} \cdot a + \frac{r-1}{n} \cdot c((r-1) + \frac{n-r}{n} \cdot c \cdot (n-r) + a \cdot n \}$

$= n \sum_{r=1}^{n} \frac{1}{n} \cdot a + \frac{1}{n} \sum_{r=1}^{n} \frac{r-1}{n} \cdot c((r-1) + \frac{1}{n} \sum_{r=1}^{n} \frac{n-r}{n} \cdot c(n-r) + \frac{1}{n} \sum_{r=1}^{n} a \cdot n$

$= \frac{a}{n} + \frac{c}{n^2} \sum_{r=1}^{n} (r-1)^2 + \frac{c}{n^2} \sum_{i=1}^{n} (n-r)^2 + a \cdot n$

// middle terms are same

$= \frac{2c}{n^2} \sum_{r=1}^{n} (r-1)^2 + a \cdot n + \frac{a}{n}$

$\leq \frac{2c}{n^2} \cdot \int_0^n x^2 \, dx + a \cdot n + \frac{a}{n} = \frac{2c}{n^2} \cdot \frac{1}{3} n^3 + an + \frac{a}{n}$

$= \frac{2}{3} c \cdot n + a \cdot n + \frac{a}{n}$

$\leq c \cdot n$ when $c > 7a$.

Because $\frac{2}{3} < 1$, we can conclude i/Select to have an average-case time complexity of $O(n)$.

(5) For any given n, if you want to sort n numbers, the fastest approach would be $\lceil \log_2 n! \rceil + C$ comparisons. This, however, assumes we only compare 2 numbers at a time. If we were comparing 5 numbers in a single operation, it should follow that the fastest approach could take $\lceil \log_5 n! \rceil + C$ comparisons. This is because if we can compare 5 numbers instead of 2 numbers, each comparison will split the data set into fifths rather than halfs.