Refo Yudhanto

CS350

HW 3

1. int min, max;

array a;

check if array only have 1 data

if length==1{

max = a[0];

min = a[0];}

Initialize min and max

if a[0] > a[1] {

max = a[0];

min = a[1];}

else {

max = a[1];

min = a[0];}

start loop starting from 2

for (int i = 2; i <= a.length - 2;) {

if (a[i] > a[i + 1]) {

min = Math.min(min, a[i + 1]);

max = Math.max(max, a[i]);

} else {

min = Math.min(min, a[i]);

max = Math.max(max, a[i + 1]);

}

i = i + 2;}

  Continuation for odd number length

if a.length % 2 == 1 {

min = Math.min(min, a[a.length - 1]);

max = Math.max(max, a[a.length - 1]);}

1. Let i = 5 and n = 15. S takes O (i \* n) time, which means (5 \* 15) in this case. While T takes O (n log n) time, which mean (16 \* log 16) = (16 \* 4) in this case. Hence, the algorithm S performs better than the algorithm T when the value of i is less than the value of log n. On the other hand, Let i = 12 and n = 16. S takes O (i \* n) time to compute the result, which is equal to (12 \* 16) in this case. T takes O (n log n) time, which is equal to (16 \* log 16) =

(16 \* 4) in this case. Hence, the algorithm T performs better than the algorithm S when the value of i is greater than the value of log n.

1. When k=3, number of median & groups is n/3. Worst case scenario would be n-n/3 = 2n/3

Tw(n)<= θ(n)+Tw(n/3)+Tw(2n/3)

When k=7, number of median &groups is n/7. Worst case scenario would be n-2n/8 = 5n/7

Tw(n)<= θ(n)+Tw(n/7)+Tw(5n/7)

1. Since we knew quickselect’s worst case is O(n^2), then the worst case of ilselect is O(n^2). While the average case, Tavg(n) can be formulized as:

Tavg(n)= θ(n)+1/n\*Σ(O(1)\*1/n+O(r-1)\*((r-1)/n)+O(n-r)\*(n-r)/n)

Which then simplified

Tavg(n)=θ(n)+2/n\*Σ(1<=r<=n)O(r-1)\*((r-1)/n)

Then we can simplify and get Tavg(n)=θn

1. Since the operation compares number, so the comparison of N will be

(N-1)+(N-2)+….+1=N(N-1)/2