Ashmal Anis
1419-0305
see: 41

Algorithms Assignment

Question #01) Sort in descending order (highest → lowest)

15, 6, 12, 8, 7, 1, 9, 3, 5, 23

show the value of key variable, $O(n^2)$. use loop invariant.

① [15], 6, 12, 8, 7, 1, 9, 3, 5, 23 , K=6, it remains at that position
② 15, 6, 12, 8, 7, 1, 9, 3, 5, 23 , K=12
③ 15, 12, 6, 8, 7, 1, 9, 3, 5, 23 , K=8
15, 12, 8, 6, 7, 1, 9, 3, 5, 23 , K=7
15, 12, 8, 7, 6, 1, 9, 3, 5, 23 , K=1
15, 12, 8, 7, 6, 1, 9, 3, 5, 23 , K=9
15, 12, 9, 8, 7, 6, 1, 3, 5, 23 , K=3
15, 12, 9, 8, 7, 6, 3, 1, 5, 23 , K=5
15, 12, 9, 8, 7, 6, 5, 3, 1, 23 , K=23
23, 15, 12, 9, 8, 7, 6, 5, 3, 1 done.

Time complexity:-

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{j=2}^{n} (t_j) + c_6 \sum_{j=2}^{n} (t_j - 1) + c_7 \sum_{j=2}^{n} (t_j - 1) + c_8(n-1)$$

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n+1)}{2} - 1\right) + c_7\left(\frac{n(n+1)}{2} - 1\right) + c_8(n-1)$$

$$T(n) = O(n^2)$$

Loop Invariant:-

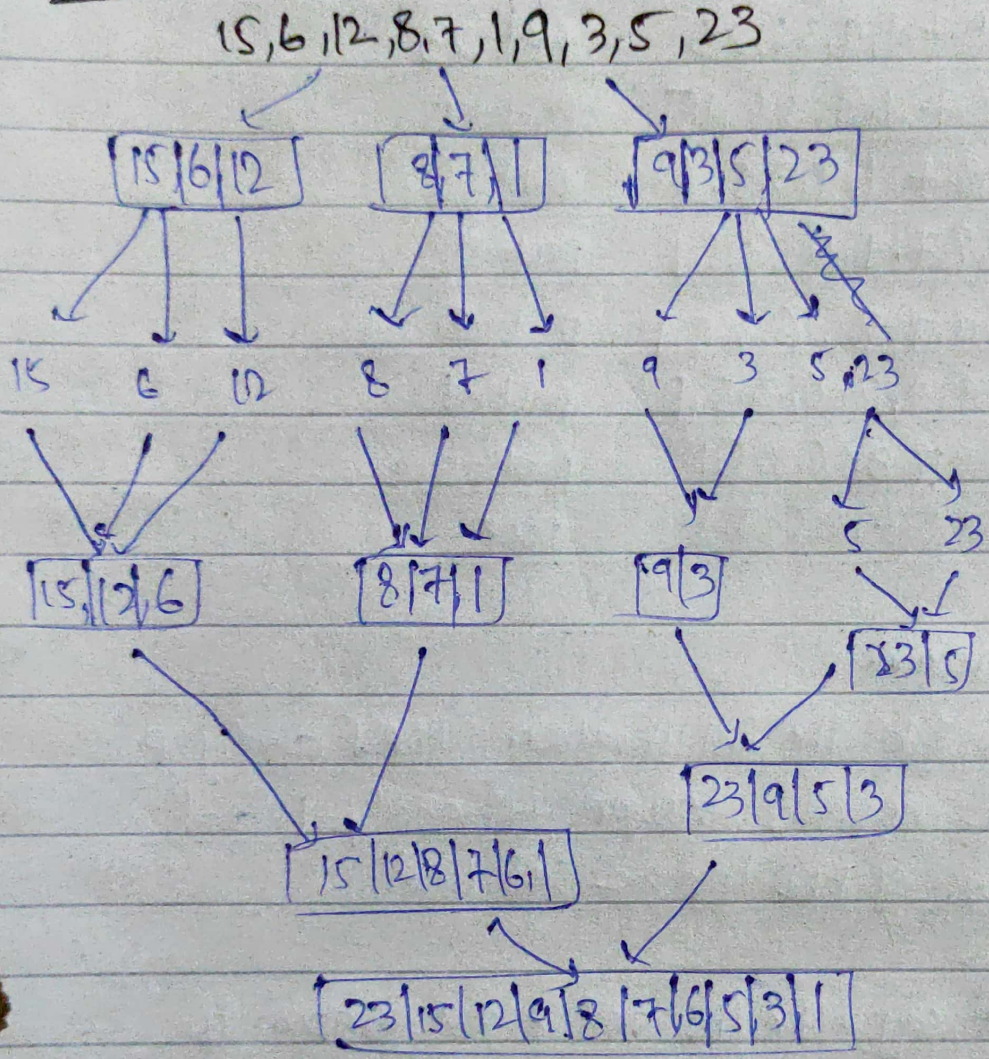
At the start of each iteration of for loop, the subarray A[1...j-1] consists of elements originally in A[1...j-1] but in sorted order.

Initialization:- At the start of for loop j=2 so the array A[1...j-1] = A[1...1] which is the first element which is already sorted element.

Maintenance:- As the iteration of loop increments imagine j=3, at the start of each iteration A[1...2], during each inner loop the element will shift towards A's sorted position in descending position, thus A[1...2] is sorted.

Termination:- At the end of for loop, the array is completely sorted in descending position. at the end j becomes n+1 i.e the j will be at a step ahead of last position and A[1...n+1-1] = A[1...n] and the complete array is sorted.

Question #2) show steps of merge sort in descending order



Time Complexity:-

$T(n) = aT(n/b) + f(n)$

$a=3, b=3, f(n) = O(n)$

$d=1, b^d = 3^1 = 3$

$T(n) = 3T(n/3) + O(n)$

$a = b^d, \quad 3=3$

$\Theta(n^d \log n)$

$\Theta(n \log_3 n)$

Question #3) Repeat for Quick Sort

15, 6, 12, 8, 7, 1, 9, 3, 5, 23 pivot = 23
23, 6, 12, 8, 7, 1, 9, 3, 5, 15 pivot = 15
23, 15, 12, 8, 7, 1, 9, 3, 5, 6 pivot = 6
23, 15, 12, 8, 7, 9, 6, 3, 5, 1 pivot = 9
23, 15, 12, 9, 7, 8, 6, 3, 5, 1 pivot = 8
23, 15, 12, 9, 7, 8, 6, 3, 5, 1 pivot = 1
23, 15, 12, 9, 8, 7, 6, 5, 3, 1 pivot = 5
23, 15, 12, 9, 8, 7, 6, 5, 3, 1 Done.

Loop Invariant correctness:

Initialization before the first iteration, i = p-1 and j = p
Because there are no values between i+1 and j-1 and first two conditions of the loop are satisfied.

maintenance: lets say after the end of 1st iteration the highest element gets on its sorted position and i becomes p and elements between i+1 and j-1 are in its right place.

Termination: At the termination, j = r. Thus, every entry is in one of the three invariants, and the array is partitioned into three sets, less than pivot, greater than pivot and the pivot.

Question #4)

O(nlogn)

If the Array has a majority element of a particular emoji, it should be the majority element in A1 or A2 or both. If both has the same number of emoji, it's the majority emoji element. This takes O(logn) time

If one of the array has majority element, then count the number of repetition in one half and the other O(n) to see if that's a majority element

we can split recursively and count which will make it $T(n) = 2T(n/2) + O(n)$, $T(n) = n\log n$.

O(n)

we will use single for loop and 2 counter variables, counter and max counter variable and everytime A[maxindex] == A[i] we will increase the count otherwise decrease the count by 1 one linear traversal to check the freaveney of the emoji type thus linear traversal of array + finding freaveney = O(n) time total.

Question#5)

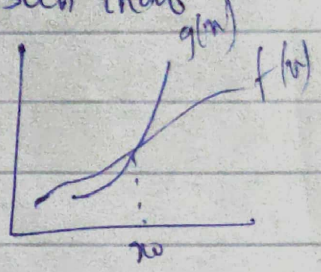
To solve this rating problem in O(nlog n) format what we can do is to first sort the complete array using any logn time sorting such as merge sort or quick sort and then simply divide the rating into even and odd indexes to evenly divide the rating.

Question # 7:

Algorithms efficiency is measured through it's scalibility and we also check space and time complexity which depends on input size.

Asymptotic Bounds.

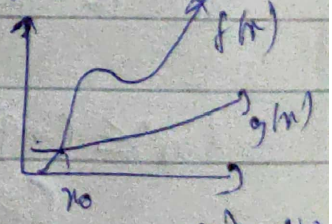
$f(n) \in O(g(n))$ if there exist $c, n_0 > 0$ such that $f(n) \leq c * g(n)$ for all $n \geq n_0$.



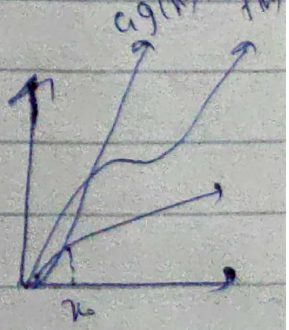
-after multiplying constant with $g(n)$, it will bound the function above $g(n)$ at some point. $f(n)$ graph does not intersect with $c * g(n)$ after point n_0.

$f(n) \in \Omega(g(n))$ if there exist $c, n_0 > 0$ such that $f(n) \geq c * g(n)$ for all $n > n_0$.

there is a constant c that $c * g(n)$ bounds $f(n)$ below from some points onwards.



$f(n) \in \Theta(g(n))$ if there exists $c_1, c_2, n_0 > 0$ such that $c_1 * g(n) \geq f(n) \geq c_2 * g(n)$ for all $n \geq n_0$ constant that bound $f(n)$ from both below and above, then $f(n)$ is in $\Theta(g(n))$



Question # 08

a) $T(n) = 4T\left(\frac{n}{2}\right) + n^3$

$a = 4, b = 2, f(n) = n^3$

$d = O(f(n)) = O(n^3) = 3$ (power of big-oh notation)

$b^d = 2^3 = 8$

$a = 4, b^d = 8$

$\therefore a < b^d \quad T(n) = \theta(n^d)$

$= \theta(n^3)$

b) $T(n) = 3T\left(\frac{n}{2}\right) + n^2$

$a = 3, b = 2, f(n) = n^2$

$O(f(n)) = O(n^2)$

$d = 2$ (power of O function)

$b^d = 2^2 = 4$

$\therefore a < b^d \quad T(n) = \theta(n^d)$

$T(n) = \theta(n^2)$

c) $T(n) = 9T\left(\frac{n}{2}\right) + n$

$a = 9, b = 2, f(n) = n$

$d = O(n) = 1$

$b^d = 2^1 = 2$

$\therefore a > b^d \quad \theta(n^{\log_b a})$

$= \theta(n^{\log_2 9})$

Question #09

a) $T(n) = 6T\left(\frac{n}{2}\right) + n, \quad T(1) = 1$ — ①

$$T(n/2) = 6T\left(\frac{n/2}{2}\right) + \frac{n}{2} = 6T\left(\frac{n}{4}\right) + \frac{n}{2}$$

$$T(n) = 6\left[6T\left(\frac{n}{4}\right) + \frac{n}{2}\right] + n$$

$$T(n) = 6^2 T\left(\frac{n}{2^2}\right) + \frac{6n}{2} + n \quad \text{— ②}$$

$$T(n/4) = 6T\left(\frac{n/4}{2}\right) + \frac{n}{4} = 6T\left(\frac{n}{8}\right) + \frac{n}{4}$$

$$T(n) = 6^2\left[6T\left(\frac{n}{8}\right) + \frac{n}{4}\right] + \frac{6n}{2} + n$$

$$T(n) = 6^3 T\left(\frac{n}{2^3}\right) + \frac{6^2 n}{2^2} + \frac{6n}{2} + n \quad \text{— ③}$$

$$T(n) = 6^K T\left(\frac{n}{2^K}\right) + \frac{6^{K-1} n}{2^{K-1}} + \frac{6^{K-2} n}{2^{K-2}} + \cdots \frac{6n}{2} + n$$

Base case: $T(1) = 1$

$$\frac{n}{2^K} = 1, \quad n = 2^K, \quad K = \log_2 n$$

$$T(n) = 6^{\log_2 n} T\left(\frac{n}{n}\right) + n\left(\frac{6^{K-1}}{2^{K-1}} + \frac{6^{K-2}}{2^{K-2}} + \cdots \frac{6}{2} + 1\right)$$

$6^{\log_2 n} T(1) + n(K+1)$

$6^{\log_2 n}(1) + (\log_2 n)n$

$n^{\log_2 6} + n\log_2 n$

$= O(n^{\log_2 6})$

b) $3 + T\left(\frac{n}{2}\right)$, $(T(1)=1)$

$T(n) = 3T\left(\frac{n}{2}\right) + 3 \longrightarrow (1)$

$T\left(\frac{n}{2}\right) = 3T\left(\frac{n/2}{2}\right) + 3 = 3T\left(\frac{n}{4}\right) + 3$

$T(n) = \left[3T\left(\frac{n}{4}\right) + 3\right] + 3 = 3T\left(\frac{n}{2^2}\right) + 3 + 3 \longrightarrow (2)$

$T\left(\frac{n}{4}\right) = T\left(\frac{n/4}{2}\right) + 3 = T\left(\frac{n}{8}\right) + 3$

$T(n) = 3\left[T\left(\frac{n}{8}\right) + 3\right] + 3 + 3$

$T(n) = 3T\left(\frac{n}{2^3}\right) + 3^2 + 3 + 3 \longrightarrow (3)$

$T\left(\frac{n}{8}\right) = T\left(\frac{n}{16}\right) + 3$

$T(n) = 3\left[T\left(\frac{n}{16}\right) + 3\right] + 3^2 + 3 + 3$

$T(n) = 3T\left(\frac{n}{24}\right) + 3^2 + 3^2 + 3 + 3 \longrightarrow (4)$

$3T\left(\frac{n}{2^k}\right) + 3 + 3 + 3 + \cdots + k$

$\sum_{i=0}^{k} 3 = \log n \qquad \frac{n}{2^k} = 1$

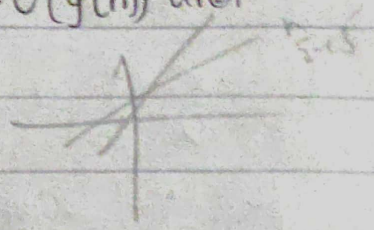
$T\left(\frac{n}{n}\right) + \log n$

$T(1) + \log n$

$= \Theta(\log n)$

Question # 10)

- for all positive $f(n)$, $g(n)$ and $h(n)$, if $f(n) = O(g(n))$ and $f(n) = \Omega(h(n))$, then $g(n) + h(n) = \Theta(f(n))$



$f(n) = \frac{n}{2} + 5$ $\quad$ $h(n) = \frac{n}{2} + 5$

$g(n) = n$

$f(n) = O(n)$ $\quad$ $f(n)$

$\frac{n}{2} + 5 = O(n)$ $\quad$ $\frac{n}{2} + 5 = \Omega(\frac{n}{2} + 5)$

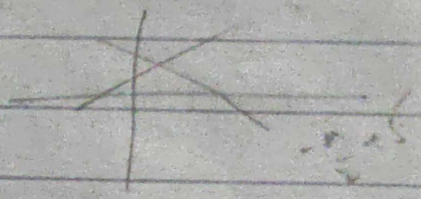
$n - \frac{n}{2} + 5 = \frac{n}{2} + 5$

$\frac{2n - n}{2} + 5 = \frac{n}{2} + 5$

True

- If $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$, $(f(n))^3 = \Theta(g(n)^3)$

$f(n) = \frac{n}{2} + 5$ $\quad$ $g(n) = n$ $\quad$ $h(n) = \frac{n}{5} + 5$



$f(n) = O(n)$

$\frac{n}{2} + 5 = O(n)$

$\frac{n}{2} + 5 = \Omega(\frac{n}{5} + 5)$

$f(n) = \Omega(g(n))$

$\frac{n}{2} + 5 = \Theta(n)$

$(\frac{n}{2} + 5)^3 = O(n^3)$

$\frac{n^3}{8} + n^2 + 3(\frac{n}{2})(5)(\frac{n}{2} + 5) = n^3$ ∴ True

if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$ $(f(n))^2 = (g(n))^2$

$f(n) = \frac{n}{2} + 5$, $g(n) = n$

$\left(\frac{n}{2} + 5\right)^2 = \frac{n^2}{4} + 5n + 25$

$(n)^2 = n^2$

False bcz $\frac{n^2}{4} + 5n + 25 < n^2$