Questions 1 : Find Third Max

Algorithm 1 :

**findThirdMax(A)**

len <- length(A)

if len < 3 then

return MIN\_VALUE

first 🡨 A[0]

firstIndex 🡨 0

secondIndex 🡨 0

for i 🡨 1 to len-1 do

if A[i] > first then

first🡨A[i]

firstIndex 🡨 i

second 🡨 MIN\_VALUE

for i 🡨 0 to len -1 do

if i ! = firstIndex AND A[i] > second AND A[i] <= first then

second 🡨 A[i]

secondIndex 🡨 i

third 🡨 MIN\_VALUE

for i 🡨 0 to len -1 do

if i ! = firstIndex AND i != secondIndex AND A[i] > third AND A[i] <= second then

third 🡨 A[i]

return third

**Numbers of operations = 23n + 8**

Algorithm 2 : **findThirdMax(A)**

|  |  |
| --- | --- |
|  | **# Operations** |
| max 🡨 MIN\_VALUE | 1 |
| preMax 🡨 MIN\_VALUE | 1 |
| prePreMax 🡨 MIN\_VALUE | 1 |
|  |  |
| for num in input | n + 1 + n(increments) |
| if num > max then | n |
| prePreMax 🡨 preMax | n |
| preMax 🡨 max | n |
| max 🡨 num | n |
|  |  |
| else if num > preMax then | n |
| prePreMax🡨 preMax | n |
| preMax 🡨 num | n |
|  |  |
| else if num > prePreMax then | n |
| prePreMax 🡨 num | n |
|  |  |
| return prePreMax | 1 |
|  | **Total = 6n + 5** |

O(n)

Algorithm 3 :

getThirdMax(A)

len 🡨 length(A)

if len = 0 then return MIN\_VALUE

A.sort()

if len < 3 then return A[len – 1]

else return A[len-3]

O(nlogn)

Question 2.

|  |  |
| --- | --- |
| **Function** | **Θ-Notation** |
| 1,10 | Θ(1) |
| log(log n) | Θ(log log n) |
| log n | Θ(log n) |
| ln n | Θ(log n) |
| n^(1/k) (k>3) | Θ(n^(1/k)) |
| n^(1/3) | Θ(n^(1/3)) |
| n^(1/3) log n | Θ(n^(1/3) log n) |
| n^(1/2) | Θ(n^(1/2)) |
| n^(1/2) log n | Θ(n^(1/2) log n) |
| n log n | Θ(n log n) |
| n^2 | Θ(n^2) |
| n^3 | Θ(n^3) |
| n^k (k>3) | Θ(n^k) |
| 2^n | Θ(2^n) |
| 3^n | Θ(3^n) |
| n! | Θ(n!) |
| n^n | Θ(n^n) |