Q1:

A1:

Algorithm arrayThirdMax(A, n)

Input array A of n integers (n >= 3)

Output 3rd maximum element of A

	Operations
currentMax <- A[0]	**2**
// index of max value	
max1Index <- 0	**1**
temp <- currentMax	**1**
max2Index <- 0	**1**
m <- n – 1	**2**
for i <- 1 to n - 1 do	**n + 1**
if A[i] > currentMax then	**2(n - 1)**
currentMax <- A[i]	**2(n - 1)**
max1Index <- I	**(n - 1)**
else	
// assign the value which is smaller than max to temp	
temp <- A[i]	**2(n - 1)**
// the index of second max value	
max2Index <- I	**(n - 1)**

{ increment counter i }   **2(n - 1)**

// currentMax has the value which is less than the max

currentMax <- temp   **1**

//second loop

for i <-1 1 to m do   **(n - 1)**

  // skip the max value in the array

  if i != max1Index then   **(n - 1)**

   if A[i] > currentMax then   **2(n - 1)**

    currentMax <- A[i]   **2(n - 1)**

    max2Index <- I   **(n - 1)**

   else

   // assign the value which is smaller than second max value to temp

    temp <- A[i]   **2(n - 1)**

   { increment counter i }   **(n - 1)**

// 3rd loop

for i<-1 to m do   **(n - 1)**

  // skip the max and 2nd max value in the array

  if i != max1Index and i != max2Index and A[i] > temp **4(n-1)**

   temp <- A[i]   **2(n-1)**

{ increment counter i }   (n-1)

// temp is the 3rd max value in the array

return temp;   1

                                            Total   29n-17

A2:

Algorithm arrayThirdMax(A, n)

Input array A of n integers (n >= 3)

Output 3rd maximum element of A

                                                Operations

Max <- A[0]   **2**

PreMax <- Max   **1**

PrepreMax <- Max   **1**

if Max < A[1] then   **2**

  Max <- A[1]   **2**

else

  PreMax <- A[1]   **2**

   PrepreMax <- Max   **1**

If Max < A[2] then   **2**

  PreMax <- Max   **1**

Max <- A[2]	**2**
else	
if A[2] > PreMax then	**2**
PreMax <- A[2]	**2**
else	
PrepreMax <- A[2]	**2**
m < n-1	**2**
for i <- 3 to n - 1 do	**2(n-3)**
if A[i] > Max then	**2(n-3)**
PrepreMax <- PreMax	**(n-3)**
PreMax <- Max	**(n-3)**
Max <- A[i]	**2(n-3)**
else if A[i] > PreMax then	**2(n-3)**
PrepreMax <- PreMax	**(n-3)**
PreMax <- A[i]	**2(n-3)**
else if A[i] > PrepreMax then	**2(n-3)**
PrepreMax <- A[i]	**2(n-3)**
{ increment counter i }	**2(n-3)**
return PrepreMax	**1**
Total	**19n-32**

A3:

Algorithm arrayThirdMax(A, n)

Input array A of n integers (n >= 3)

Output 3rd maximum element of A

	Operations
A.sort()	**nlogn**
return A[lengthofA - 3];	**2**
	Total **nlogn+2**

Algorithm	Steps	Time Complexity
A1	29n – 17	O(n)
A2	19n – 32	O(n)
A3	nlogn + 2	O(nlogn)

Q2:

10,1	$\Theta(1)$
$\log(logn)$	$\Theta(\log(logn))$
$logn, lnn$	$\Theta(logn)$
$n^{1/k}$(k>3)	$\Theta(n^{1/k})$
$n^{1/3}$	$\Theta(n^{1/3})$
$n^{1/2}$	$\Theta(n^{1/2})$
$n^{1/3}logn$	$\Theta(n^{1/3}logn)$
$n^{1/2}logn$	$\Theta(n^{1/2}logn)$
$nlogn, \log n^n$	$\Theta(nlogn)$
n^2	$\Theta(n^2)$
n^3	$\Theta(n^3)$
n^k(k>3)	$\Theta(n^k)$
2^n	$\Theta(2^n)$
3^n	$\Theta(3^n)$
$n!$	$\Theta(n!)$
n^n	$\Theta(n^n)$