2018ICSE0621
Saikpa

Date	
Page	

Sai Ram. K 5-CSE-10.
2018ICSE0621
08/01/2021

Part-B

Q.1] **Quicksort :-**

Quicksort is a divide and conquer algorithm. It picks an element as pivot and partitions the given array around the pivot.

→ **ALGORITHM:-**

Step1:- Partitioning the array into 2.

```
partition(a, low, high)
{
    pivot = a[low], i=low+1, j=high
    while(pivot > a[i])
        i++
    while(pivot < a[j])
        j--
    if(i<j)
        swap(a[i], a[j])
    else
        swap(a[low], a[j])
    return j
}
```

Step 2: Sorting the array

```
qsort(a, low, high)
{
    if(low<high)
    {
        j= partition(a, low, high)
```

2018ICSE0621

Saifaly

q sort (a, low, j-1)
q sort (a, j+1, high)
}

→ Efficiency of Quicksort :-

* Best & average case ⇒ T(n)

$$T(n) = \begin{cases} T(1) = 0, & \text{if } n = 1 \\ \underbrace{T(\tfrac{n}{2})}_{\text{Time taken for left pivot}} + \underbrace{T(\tfrac{n}{2})}_{\text{Time for right pivot}} + \underbrace{n}_{\text{Time for partitioning}}, & \text{otherwise} \end{cases}$$

$T(n) = 2T(\frac{n}{2}) + n$

Using Master's theorem,

$a = 2 \quad b^k = 2^1$

$\boxed{a = b^k} \qquad \boxed{k = 1}$

$\therefore T(n) = \theta(n^k \log n)$

$= \theta(n^1 \log n)$

$\boxed{\therefore T(n) = \theta(n \log n)}$

Average & Best case have same complexity

$\boxed{\therefore T(n) = \Omega(n \log n)}$

* Worst case :-

$T(n) = T(0) + T(n-1) + n$

$$T(n) = \begin{cases} T(1) = 0, & \text{if } n = 1 \\ T(n-1) + n, & \text{otherwise} \end{cases}$$

We proceed further,

$T(n) = T(n-1) + n$ --- (1)

Replace n by n-1 in (1)

2018ICSE0621

Saifyna

Date

Page

$= T(n-1-1) + n-1+n$

$= T(n-2) + (n-1)+n$

$= T(n-3) + (n-2)+(n-1)+n$

$\vdots$

$= T(n-n) + (n-(n-1)) + (n-(n-2))+n$

$= T(0) + (\cancel{n}-\cancel{n}+1) + (\cancel{n}-\cancel{n}+2)+n$

$= 0+1+2+\ldots+n$

$= \frac{n(n+1)}{2} = \frac{n^2}{2}+\frac{n}{2} \approx n^2$

We consider the highest power,

Hence, $\boxed{T(n) = O(n^2)}$

→ Differences between Merge Sort & Quick Sort.

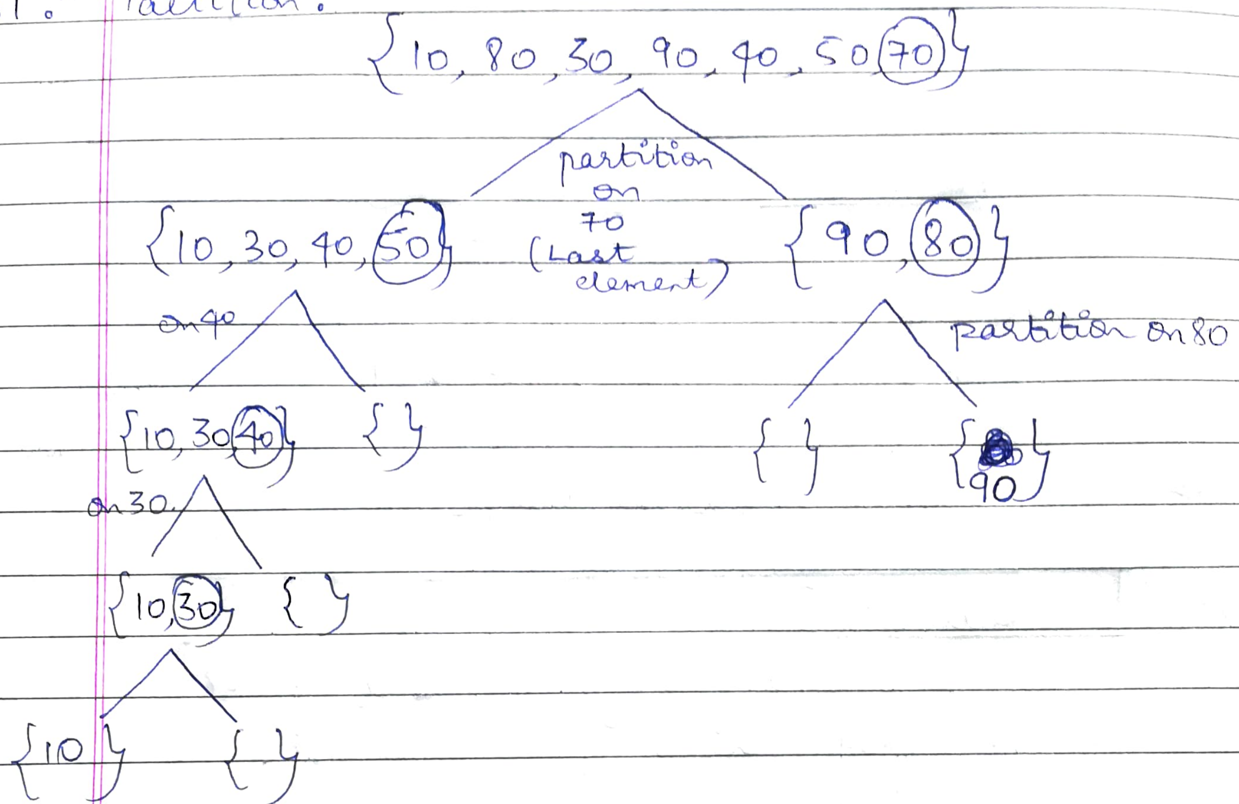
	Quick Sort	Merge Sort.
(i)	The splitting of array is in any ratio not exactly into half.	Splitting of array is also not necessarily half but uses different approach.
(ii)	Worst case complexity is $O(n^2)$	Worst case complexity is $O(n \log n)$
(iii)	Works well on smaller arrays	Works good with any size.
(iv)	Works faster than other algorithms for small dataset.	It has consistent speed on any data.
(v)	It is not stable	It is stable
(vi)	Preferred for arrays	Preferred for Linked lists
(vii)	Method of sorting is internal	Method of sorting is external.

2018ICSE0621
Sai

→ Example for quicksort :

Consider {10, 80, 30, 90, 40, 50, 70}

Step1 :- Partition :



Step2 :- Merging Back we get :-

{10, 30, 40, 50, 70, 80, 90}

[Merging is done by traversing.]