

time for diff values of n.

// static \rightarrow need not create objects
(available to all the objects).

classmate

Date

Page

P4) A)

Sort the given set of n integer elements using quick sort method and compute its time complexity. Run the program for varied values of n > 5000 & record the time taken to sort. Plot a graph of the time taken versus n graph sheet. The elements can be read from a file & can be generated using the random number generator. Demonstrate using Java how the divide & conquer method works along with its time complexity analysis: worst, avg, best cases.

import java.util.Random;
import java.util.Scanner;

for generating no. & randomly

public class quick {
static int a[] = new int[100000];

public static void main (String[] args) throws
ArrayIndexOutOfBoundsException

{
Scanner in = new Scanner(System.in);
long start, end;

System.out.println("-- Quick Sort --");

System.out.println("Enter the number of
Elements to be sorted");

(I can create one more class, explain quick sort & data object & then the other class & data object & then the method.)

// Name time \rightarrow fun
// Attention time.


```

int n = rn.nextInt();
Random rand = new Random();
for (int i = 0; i < n; i++)
    a[i] = rand.nextInt(100);
System.out.println("Array elements to be sorted:");
for (int i = 0; i < n; i++)
    System.out.println(a[i] + " ");
int low = 0, high = n - 1;
a[n] = 999;
start = System.nanoTime();
int m = partition(a, low, high);
quicksort(a, low, high);
end = System.nanoTime();
System.out.println("The sorted array is:");
for (int i = 0; i < n; i++)
    System.out.println(a[i] + " ");
System.out.println("In the time taken to sort
    is: " + (end - start) + " ns");
System.out.println("***\n\n");

```

```

}

static void quicksort (int a[], int low, high)
{
    int mid;
    if (low < high)
    {

```



```
mid = partition(a, low, high);  
quicksort(a, low, mid-1);  
quicksort(a, mid+1, high);
```

}

}

```
static int partition(int a[], int low, int high)
```

{

```
    int pivot = a[low];
```

```
    int i = low+1;
```

```
    int j = high;
```

```
    while(i <= j)
```

{

```
        while(a[i] <= pivot)
```

{

```
            i++;
```

}

```
        while(a[j] > pivot)
```

{

```
            j--;
```

}

```
    if(i < j)
```

{

```
        swap(a, i, j)
```

}

}

```
swap(a, low, j);  
return j;
```

```
y  
static void swap (int a[], int i, int j)  
{  
    int temp;  
    temp = a[i];  
    a[i] = a[j];  
    a[j] = temp;  
y
```

y

Output:

-- Quick Sort --

Enter the number of Elements to be sorted:

9

Array Elements to be sorted are:

55

10

77

12

78

16

17

38

14

The sorted array is:

10

12

14

16

17

38

55

77

78

the time taken to sort the array is: 15100ms

Array:

0	1	2	3	4	5	6	7	8
55	10	77	12	78	16	17	38	14
↑ pivot	↑ p							↑ q

 $q \leq p \Rightarrow \text{true}$ 1) $a[p] \leq 55$ $10 \leq 55$ $p++$ 2) $a[q] \leq 55$ $77 \leq 55 \rightarrow \text{false}$ $a[p] > 55$ $14 > 55 \rightarrow \text{false}$

2

 $l = 2$ & $p = 8$ $2 < 8$ swap $a[l]$ & $a[p]$

array \rightarrow

0	1	2	3	4	5	6	7	8
55	10	14	12	38	16	17	38	77
\uparrow p		\uparrow q						\uparrow q

$$p \leq q \Rightarrow 2 \leq 8 \Rightarrow \text{true}$$

$$3) a[p] \leq 55 \quad 4) a[3] \leq 55 \quad 5) a[4] \leq 55$$

$$14 \leq 55$$

$$12 \leq 55$$

$$38 \leq 55$$

$$p++$$

$$i++$$

$$q \text{ balu}$$

$$p=4$$

$$\Rightarrow a[q] > 55 \quad 6) a[7] > 55$$

$$77 > 55$$

$$38 > 55$$

$$q--$$

$$q \text{ balu}$$

$$p=4 \quad q=7$$

$$p < q \Rightarrow 4 < 7$$

$$\text{swap } a[p] \text{ \& } a[q]$$

array \rightarrow

0	1	2	3	4	5	6	7	8
55	10	14	12	38	16	17	78	77
\uparrow p				\uparrow q			\uparrow q	

$$p \leq q \Rightarrow 4 \leq 7 \Rightarrow \text{true}$$

$$7) a[4] \leq 55 \quad 8) a[5] \leq 55 \quad 9) a[6] \leq 55$$

$$38 \leq 55$$

$$16 \leq 55$$

$$17 \leq 55$$

$$i++$$

$$i++$$

$$i++$$

$$10) a[7] \leq 55$$

$$78 \leq 55$$

$$q \text{ balu}$$

$$p=7$$

$$p < q$$

$$7 \neq 6$$

$$\Rightarrow a[p] > 55$$

$$78 > 55$$

$$q--$$

$$q=6$$

$$a[6] > 55$$

$$17 > 55$$

$$q \text{ balu}$$

Swap pivot / low & $a[p]$, return j

array \Rightarrow 11 10 14 12 38 16 55 78 77 99
 l $mid(j)$ h

$l < h$

* $quicksort(a, l, mid-1)$ + $quicksort(a, mid+1, h)$

0 1 2 3 4 5
 11 10 14 12 38 16
 $p \uparrow$ $q \uparrow$ j
 $q \leq j \Rightarrow 1 \leq 5 \checkmark$

1) $a[p] \leq 11$ 2) $a[2] \leq 11$ 3) $a[3] \leq 11$
 $10 \leq 11$ $14 \leq 11$ $12 \leq 11$
 $p++$ $p++$ $p++$

4) $a[4] \leq 11$ $p=4$ \rightarrow $q < j$
 $38 \leq 11 \rightarrow \text{false}$ $4 < 5$
 $a[j] > 11$ swap $a[p]$ & $a[j]$
 $16 > 11$
 $\hookrightarrow \text{false}$ $j=5$

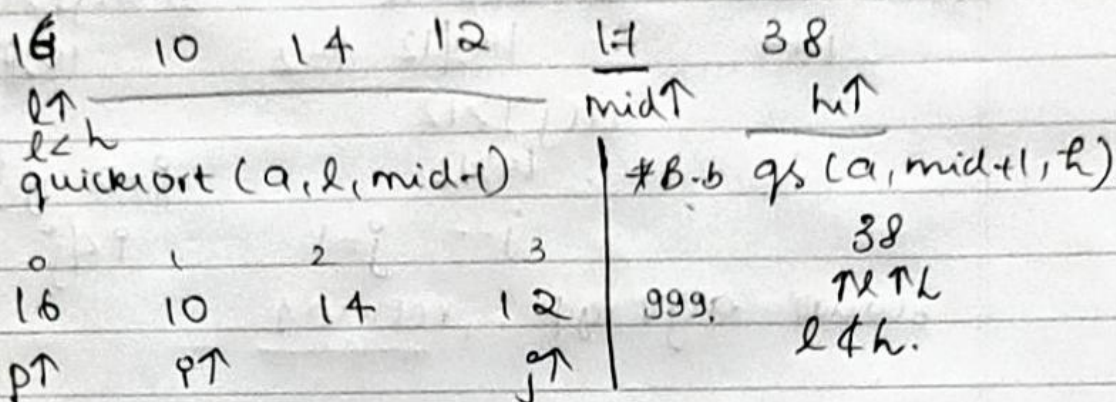
array \Rightarrow 11 10 14 12 16 38
 p q j

5) $a[p] \leq 11$ 6) $a[p] \leq 11$ $a[p] > 11$
 $16 \leq 11$ $a[5] \leq 11$ $38 > 11$
 $p++$ $38 \not\leq 11$ $p=5$ $j--$

$$\begin{array}{ll}
 1) \ a[p] > 17 & p < j \\
 16 \neq 17 & 5 \neq 4 \\
 j = 4 & 8
 \end{array}$$

swap pivot & $a[j]$, return 4

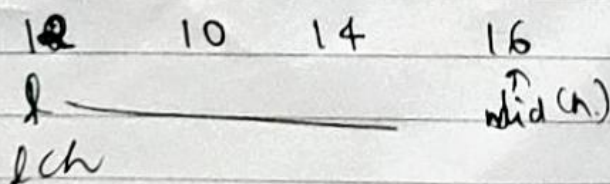
array \Rightarrow



$$\begin{array}{lll}
 1) \ a[p] \leq 16 & 2) \ a[j] \leq 16 & 3) \ a[3] \leq 16 \\
 10 \leq 16 & 14 \leq 16 & 12 \leq 16 \\
 p++ & p++ & p++
 \end{array}$$

$$\begin{array}{ll}
 p < j & p \neq j \\
 a[3] > 16 & \text{Swap pivot \& a[j]} \\
 12 \neq 16 & \text{return 3.} \\
 p = 3 \quad j = 3 &
 \end{array}$$

array \Rightarrow



Ans a) $qs(a, l, mid-1)$

0	1	2
12	10	14
$p \uparrow$	$q \uparrow$	$j \uparrow$

2) $a[p] \leq 12$
 $10 \leq 12$
 $i++$

2) $a[q] \leq 12$
 $14 \leq 12$ $p=2$

3) $a[j] > 12$
 $10 > 12$

$a[j] > 12$
 $14 > 12$

$j--$ $j=1$

$p=2$ & $j=1$
 $p < j$

swap $a[j]$ & p , return 1

*c) array

10	12	14
l	\uparrow mid	h

c.a) $qs(a, l, mid-1)$

10
 $l \uparrow h \uparrow$
 $l < h$

c-b) $qs(a, mid+1, h)$

14
 $l \uparrow h \uparrow$
 $l < h$

$b \Rightarrow$

10 12 14 16 17 38 55

7 8

*B 78 77

$\begin{matrix} \uparrow \\ \text{mid} \end{matrix}$ $\begin{matrix} \uparrow \\ \text{mid} \end{matrix}$

$\begin{matrix} \uparrow \\ \text{lt} \end{matrix}$ $\begin{matrix} \uparrow \\ \text{gt} \end{matrix}$

1) $\text{arr}[i] \leq \text{arr}[j]$

~~78 < 77~~

~~arr[j] > arr[i]~~

$77 \leq 78$

$i++$

$p =$

2) $i \neq j$

swap arr[j] & pivot

77

array 77 78.

final array:

10 12 14 16 17 38 55 77 78