2141016047 Branch: CS

Q1 Assignment-3 DL

Name: Soumyadeep Panda Regd.No: 2141016047 Sec: F Branch: CSE
Sl.No: 29

AD-1 Assignment-V

Q1

```
Partition(){
        while j < r
            iff [arr[j] < ext
                arr = swap(arr, i, j)
            i++
            j++
        arr = swap(arr, i, r)
        return i
        if (l <= r) do
        pivot = random()%n
        arr = swap(arr, pivot+1, r)
        PI = (arr, l, r)
        if (PI == K)
            b = arr(PI)
            if (a) = -1)
                            return min Value
    return Median
```

Time Complexity: O(n)

The time complexity of the original partition algo, is also O(n), same as the new one.

...ne: Saumyadeep Panda Sec: F

...ent-3 DL

A = {8, 26, 7, 18, 6, 12, 2, 5, 22, 10}

a)

8	26	7	18	6	12	2	5	22	10
Piv	P							q ← q	

b) 8 5 7 18 6 12 2 26 22 10

Piv P → p q ← q

c) q ←

8 5 7 2 6 12 18 26 22 10

Piv p → p q

d) 6 5 7 q 2 $\boxed{8}$ 12 18 26 22 q 10

Piv P → p Piv p

e) q ← q q ← q

6 5 2 7 $\boxed{8}$ 12 10 26 22 18

p → p p → p

f) 2 5 $\boxed{6}$ 7 $\boxed{8}$ 10 $\boxed{12}$ 26 22 q 18

p → p

g) 2 5 $\boxed{6}$ 7 $\boxed{8}$ 10 $\boxed{12}$ 18 22 $\boxed{26}$

now the array is sorted

no of swapping = 6

no of comparisions = 15

Q2

The average case time complexity of Quick Sort is O(n)

..mo Panda ...ment-V DL

The Worst case time complexity of Quick sort is = $O(n^2)$

The Worst case space complexity of Quick sort is = $O(\log n)$

Though the worst case time complexity of Bubble, selected and insertion sort is the same as Quick Sort, for sorting larger array Quick Sort is preferred as Quick sort follows divide & Conquer strategy, it divides the problem into Sub parts while sorting the array which requires comparitively less time than other.

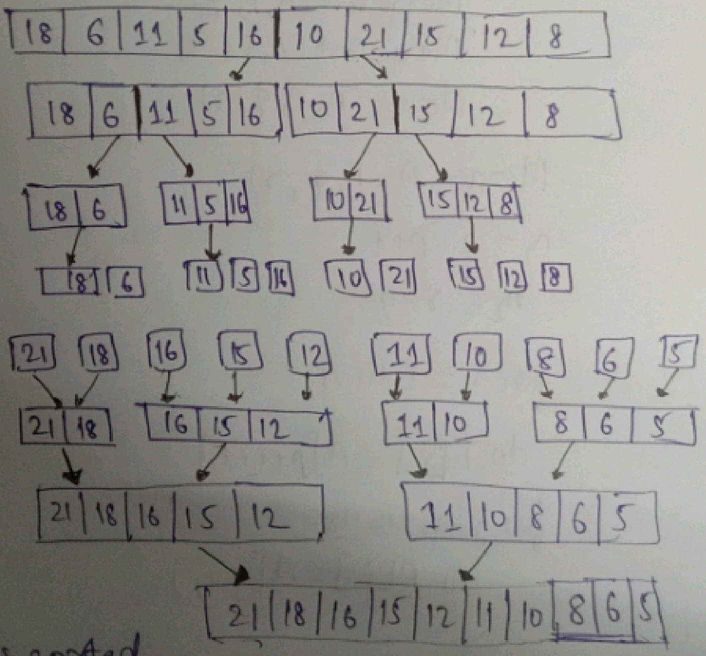
Q3

```
Merge Sort (A, P, r)
  if P < r
        then q = ⌊(P+r)/2⌋
            Merge Sort (A, P, q)
            Merge Sort (A, r, r)
            Merge (A, P, q, r)

    Merge (A, P, q, r)
    n1 = q - P + 1
     n2 = r - q
    for i = 1 to n1
     do L[i] = A[p+i+1]
    for j = 1 to n2
         do R[j] = A[q+j]
```

```
L[n1+1] = ∞
R[n1+1] = ∞
i = 1 , j = 1
for K = p to r
    if L[i] > R[j]
        ~~then A[K] = R[j]~~
        ~~j~~
        then A[K] = R[j]
             j = j+1
        else A[K] = L[i]
             i = i+1
```

A = {18, 6, 11, 5, 16, 10, 21, 15, [illegible], 12, 8}



The array is sorted

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}
A	7	10	12	8	2	1	6	3	5	4	11	9
B	3	1	7	11	8	2	6	9	5	12	4	10
C	6	2	4	11	10	12	3	8	1	5	9	7

Degree of dissimilarity of A = 5

Degree of dissimilarity of B = 3

Degree of dissimilarity of C = 4

∴ 'A' has the highest degree of dissimilarity from others, so, 'A' has the chance to win the prize.

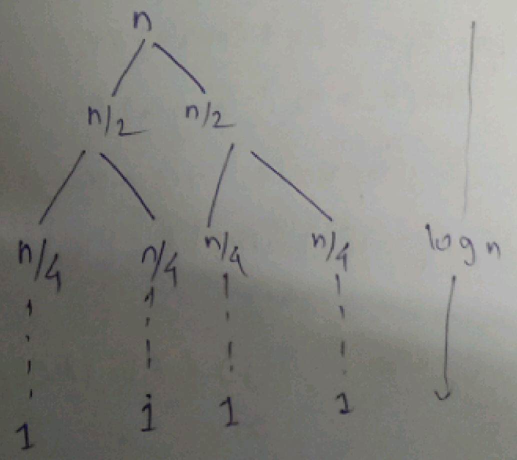
```
Preference Comparision(A[], a, B[], b, C[], c){
    a=b=c=0
    for (i=0 to A.length) do
      for (j=0 to B.length) do
        for(k=0 to C.length) do
          Compare the three arrays
         if A[i] < B[j] & C[k]
              a++
          else if B[j] < A[i] & C[k]
              b++
          else if C[k] < A[i] & B[j]
              c++
return a, b, c
```

DL

Q5

```
i> Closest Point () {
    s1 = closest Point (left half)
    min = d
    s2 = closest Point (Right half)
    Array.sort (strip, 0, size)
    for i = 0 to (i < size)
    for j = (i+1) to (j < size & strip < min)
        if (distance between strip[i] < strip[j] < min)
            min = distance between (strip[i], strip[j])
    ~~T(n) = 2T(n/2) +~~ Of
    return min.
```

$T(n) = 2T(n/2) + O(n\log n)$



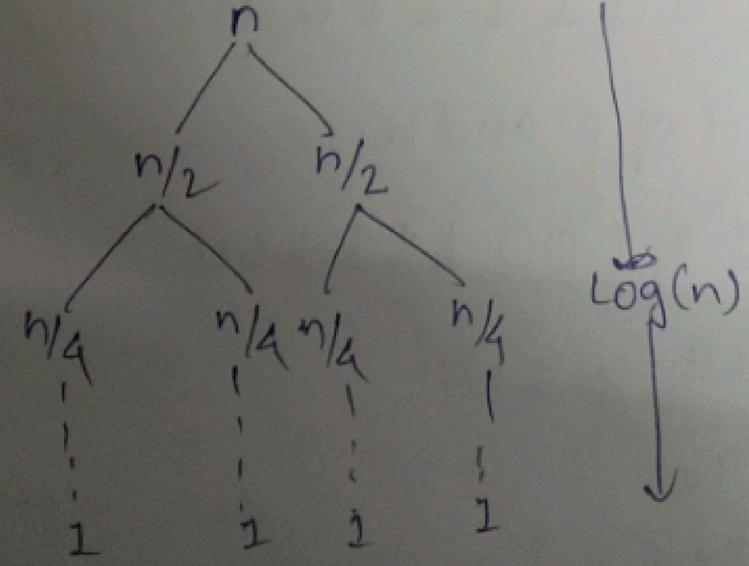
$\therefore T(n) = T(n \times \log n \times \log n)$

$= O(n \log^2 n)$

<ii> Closest Pair (P_1, P_2 ------ P_n) {

```
1. Compute sepration line L such that half the pts are
   on one side & half on other
2.   $S_1$ = Closest Pair(left half)
3.   $S_2$ = Closest Pair(Right half)
4      $\delta$ = min($S_1$, $S_2$)
5.   delete all pts further than $\delta$ from L
6.   Sort remaining points & merge them
7.   Scan points & compare distance between each
     point and then
8.      if there is any distance less than S
9.      update S
10.  return S
```

$$T(n) = 2T(n/2) + O(n \log n)$$



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

Q6

a = 10110101

b = 11001100

```
   1 0 1 1 0 1 0 1
 × 1 1 0 0 1 1 0 0
 -----------------
   0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
   0 1 1 1 0 1 0
```

```
         1 0 1 1  0 1 0 1
       × 1 1 0 0  1 1 0 0
       -------------------
          0 0 0 0  0 0 0 0
        0 0 0 0 0  0 0 0
      1 0 1 1 0 1 0 1
    1 0 1 1 0 1 0 1
   0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0
 1 0 1 1 0 1 0 1
1 0 1 1 0 1 0 1
--------------------------
1 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0
```

The product of a and b is

1000111100111100

Reversive Multiply(x, y):

1. $x = x_1 \cdot 2^{n/2} + x_0$

2. $y = y_1 + 2^{n/2} + y_0$

47 Search

```
3. Compute $x_1 + x_0$ & $y_1 + y_0$
4. P = ~~Recu~~ Recursive Multiply $(x_1 + x_0, y_1 + y_0)$
5. $x_1 y_1$ = Recursive Multiply $(x_1, y_1)$
6. $x_0 y_0$ = Recursive Multiply $(x_0, y_0)$
7. return $x_1 y_1 2^n + P(P - x_1 y_1 - x_0 y_0) 2^{n/2} + x_0 y_0$
```

Time Complexity

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + T(1 + \lceil n/2 \rceil) + \Theta(n)$$

$$T(n) = O(n^{\log 3})$$

=>

=> $T(n) = O(n^{1.585}) \approx O(n^2)$

<u>Q8</u>

$x = 10$

$P(10) = 21034$

$Q(10) = 352$

$P(x) = [4, 3\ 0\ 1\ 2]$

$Q(x) = [2\ 5\ 3\ 0\ 0]$

=> $[4\ 3\ 0\ 1\ 2]\ [2\ 5\ 3\ 0\ 0]$

=> $[200] \times [4\ 3\ 0\ 1\ 2] + [50] \times [4\ 3\ 0\ 1\ 2]$

$+ [3] \times [4\ 3\ 0\ 1\ 2]$

Name:
SL·No: 29

Assignment-3 DL

[0 0 2 0 1 3 6] +

[0 0 7 1 5 0 1] +

[8 6 0 2 4 0 0]

[8 6 9 3 0 4 7]

= 7,403,968