DESIGN & ANALYSIS OF ALGORITHMS
BCSC0012

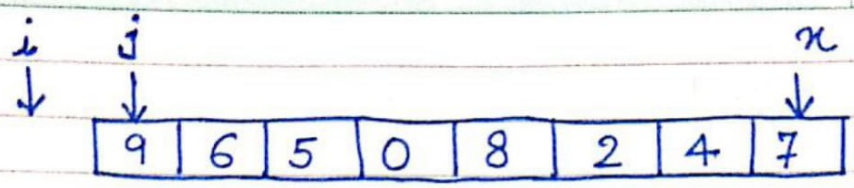
LECTURE 11 : QUICK SORT

- NISHTHA PARASHAR
ASSISTANT PROFESSOR, GLA UNIVERSITY, MATHURA

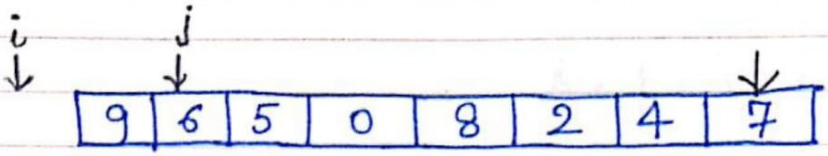
QUICK SORT

```
PARTITION (A, p, r)
{
  x = A[r]
  i = p-1
  for (j = p to r-1)
  {
    if (A[j] ≤ x)
    {
      i = i+1
      exchange A[i] with A[j]
    }
    else j = j+1
  }
  exchange A[i+1] with A[r]
  return i+1;
}
```

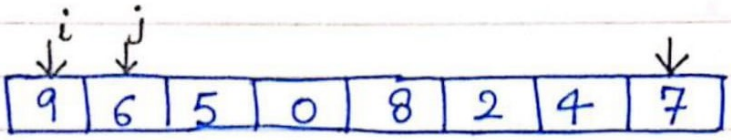
(1)



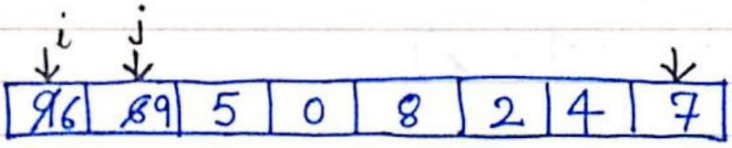
(2)



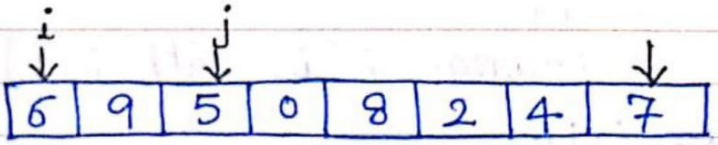
(3)



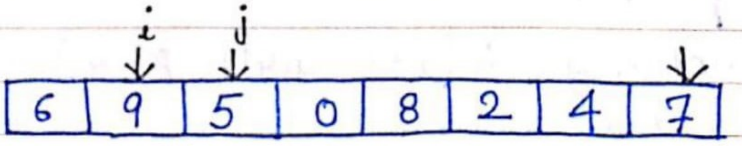
(4)



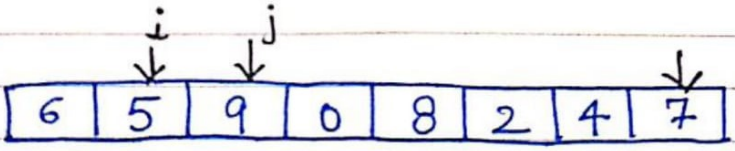
(5)



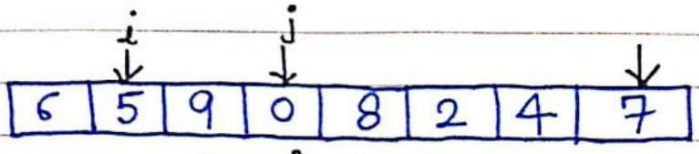
(6)



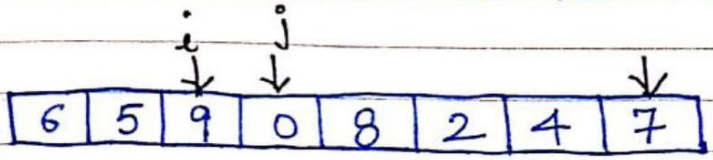
(7)



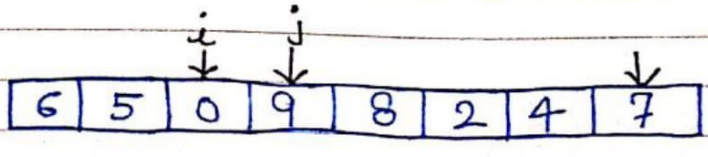
(8)

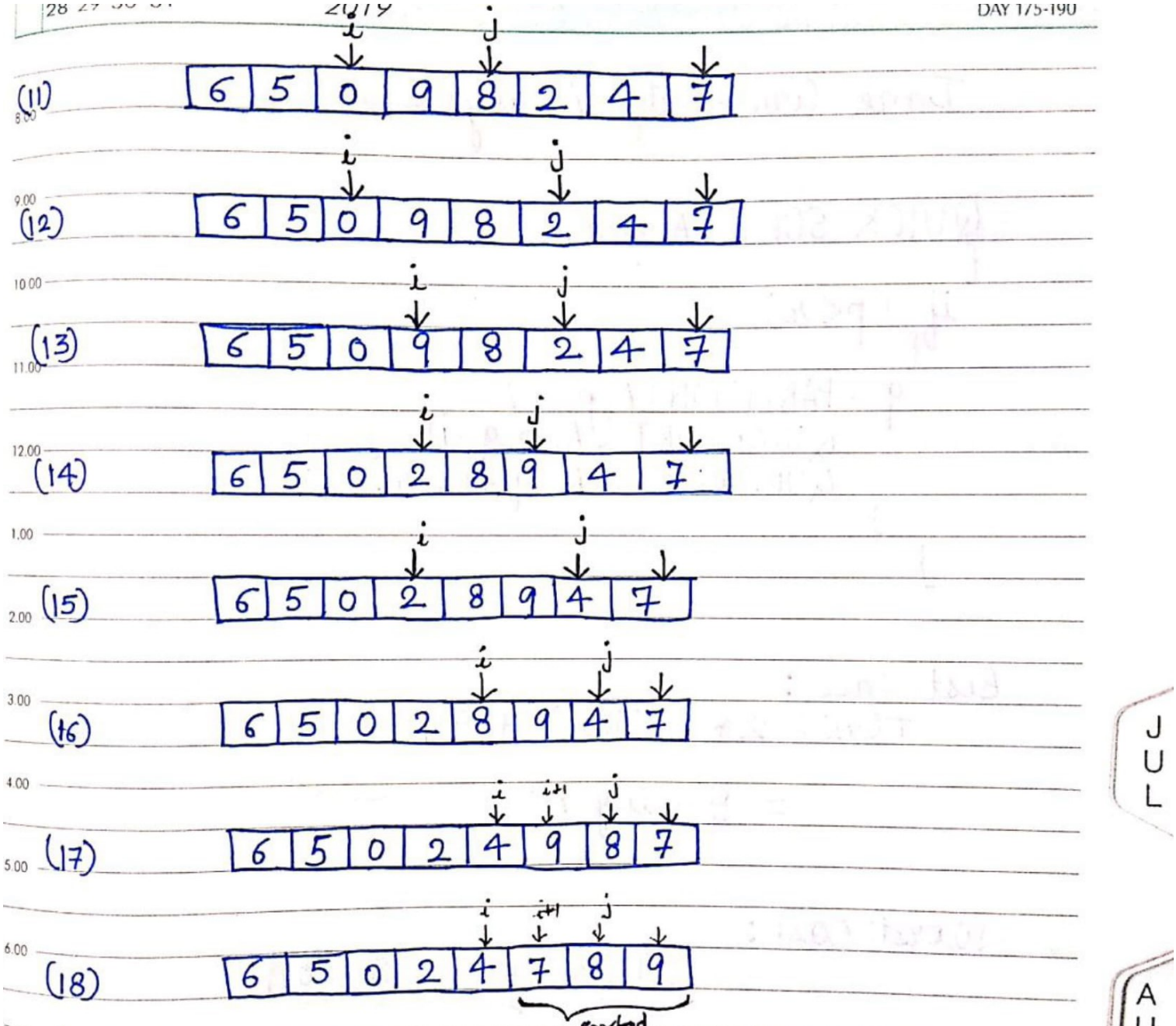


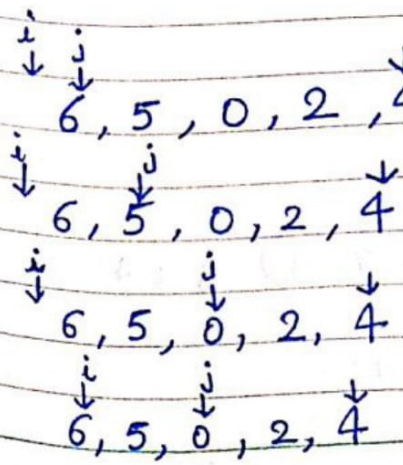
(9)

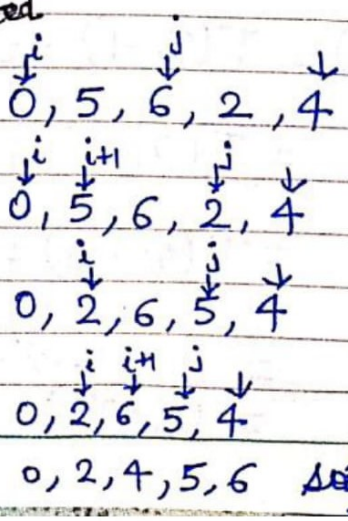


(10)




(11)
6 5 0 9 8 2 4 7
(12)
6 5 0 9 8 2 4 7
(13)
6 5 0 9 8 2 4 7
(14)
6 5 0 2 8 9 4 7
(15)
6 5 0 2 8 9 4 7
(16)
6 5 0 2 8 9 4 7
(17)
6 5 0 2 4 9 8 7
(18)
6 5 0 2 4 7 8 9
sorted


6, 5, 0, 2, 4
6, 5, 0, 2, 4
6, 5, 0, 2, 4
6, 5, 0, 2, 4


0, 5, 6, 2, 4
0, 5, 6, 2, 4
0, 2, 6, 5, 4
0, 2, 6, 5, 4
0, 2, 4, 5, 6 sorted.

Time Complexity Analysis

```
QUICK SORT (A, p, r)
{
  if (p<r)
  {
    q = PARTITION (A, p, r)
        QUICKSORT (A, p, q-1)
        QUICKSORT (A, q+1, r)
  }
}
```

Best Case:

$$T(n) = 2 * T(n/2) + n$$

$$= \underline{\theta(n \log n)}$$

Worst case:

[1, 5, 6, 7, 8, (10)]

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

$$= T(n-2) + c.(n-1) + cn$$

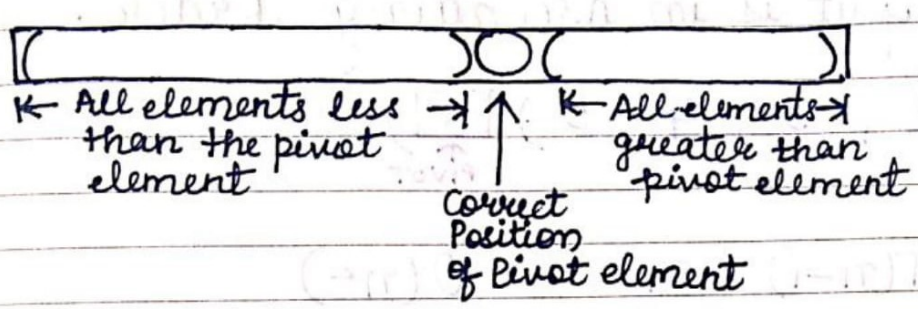
$$= T(n-3) + c.(n-2) + c.(n-1) + c.n$$

$$\vdots$$

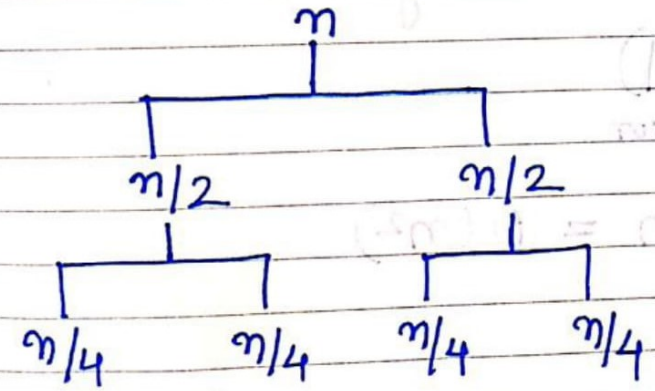
$$= c + c.2 + c.3 + \ldots\ldots + c.n$$

$$= \underline{O(n^2)}$$

QUICK SORT



→ When we have equal partitioning (BEST CASE)

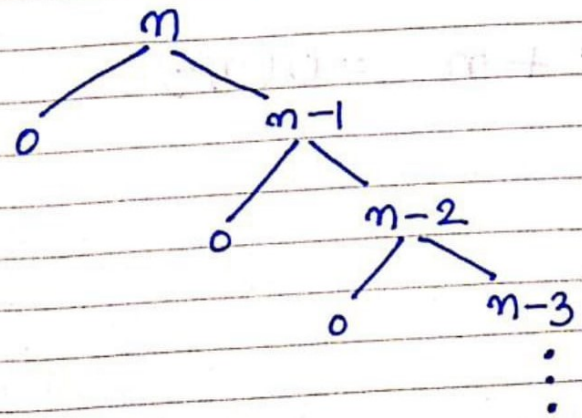


Recurrence Relation:

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

$$= \Omega(n \log n)$$

→ When we have unequal partitioning (WORST CASE)



Recurrence Relation:

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

$$= O(n^2)$$

SOME CASES :-

1. If the input is in ascending order.

((1, 2, 3, 4, 5) [6])
PIVOT

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

2. If the input is in descending order.

((6, 5, 4, 3, 2) [1])
PIVOT

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

3. Equal Elements.

((2, 2, 2, 2, 2) [2])
i j PIVOT

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