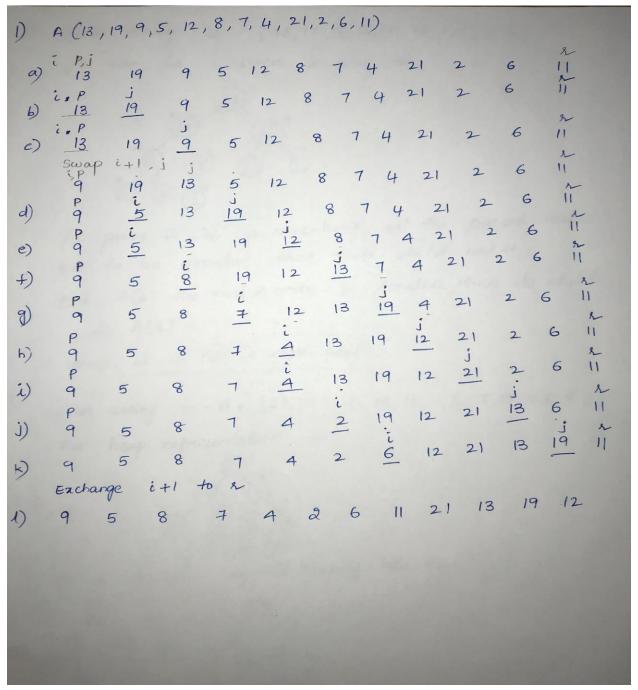
# CSCI 532 – Algorithm Design Assignment 3

Name: Badarika Namineni CWID: 50233279

**Question 1:** Using Figure 7.1 as a model, illustrate the operation of PARTITION on the array A (13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11).

# **Solution:**



**Question 2:** Give a brief argument that the running time of PARTITION on a subarray of size n is  $\Theta(n)$ .

## **Solution:**

Let us consider the algorithm for the partition

```
PARTITION (A, p, r)
```

- 1) x = A[r]
- 2) I = p-1
- 3) for (j = p to r-1)
- 4) {
- 5)  $if(A[j] \le x)$
- 6) {
- 7) i = i+1
- 8) swap A[i] with A[j]
- 9) }
- 10) }
- 11) Swap A[i+1] with r
- 12) Return i+1

The for loop makes exactly r - p iterations, each of which takes at most constant time. Since r - p is the size of the subarray, PARTITION takes at most time proportional to the size of the subarray it is called on. Hence it takes  $\Theta(n)$ .

**Question 3:** Use the substitution method to prove that the recurrence  $T(n) = T(n-1) + \Theta(n)$  has the solution  $T(n) = \Theta(n2)$ 

#### **Solution:**

This is used in worst-case scenario where one side of the partition may contain 0 elements. In this case, we get  $T(n) = T(n-1) + \Theta(n)$ 

By using substitution method,

We can write 
$$T(n) = T(n-1) + cn$$
  
 $T(n) = T(n-2) + c(n-1) + cn$  [Where  $T(n-1) = T(n-2) + c(n-1)$ ]  
 $T(n) = T(n-3) + c(n-2) + c(n-1) + cn$  [Where  $T(n-2) = T(n-3) + c(n-2)$ ]

We get, 
$$T(n) = c + c2 + c3 + cn = \Theta(n^2)$$

**Question 4:** Show that in the recurrence max  $(T(q) + T(n-q-1)) + \Theta(n)$ ,

$$0 \le q \le n-1$$
$$T(n) = \Omega(n_2)$$

# **Solution:**

$$\begin{split} T\big(n\big) &= \underset{0 \leq q \leq n-1}{max} \left(T(q) + T(n-q-1)\right) + \Theta(n) \\ &= T(n-1) + \Theta\left(n\right) \\ &= \Theta\left(n^2\right) \end{split} \tag{Where $q=0$}$$

Since it is  $\Theta$  (n<sup>2</sup>), it can also be  $T(n) = \Omega(n^2)$ 

**Question 5:** Show that quicksort's best-case running time is  $\Omega(nlgn)$ 

## **Solution:**

In the best-case, partition produces 2 sub-problems each of size not more than n/2, since one is of size [n/2] and one of size ([n/2]-1). In this case, quicksort runs much faster. The recurrence for the running time is  $T(n) = 2T(n/2) + \Theta(n)$ 

By using master's theorem, the solution for this recurrence is  $\Theta(nlogn)$ . Because it is  $\Theta(nlogn)$ , the best-case running time is  $\Omega(nlogn)$ .

**Question 6:** Show that RANDOMIZED-QUICKSORT's expected running time is  $\Omega(nlgn)$ 

## **Solution:**

Considering the lower bound of the expected running time from Lemma theorem, we get

$$E[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$$

$$\sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1}$$
Where k>=1
$$\sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{2k}$$

$$\sum_{i=1}^{n-1} \Omega(\log n)$$

We get,  $E[X] = \Omega(nlogn)$