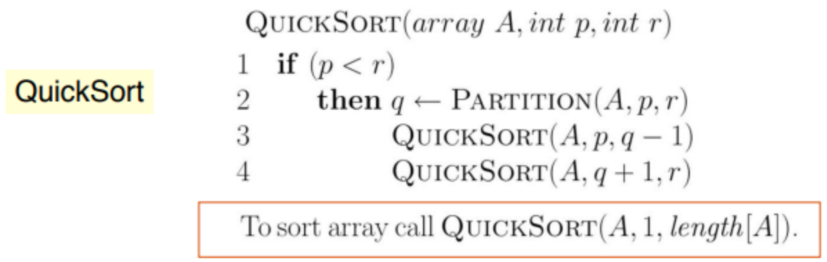
Please solve every question as a separate function with proper parameters, read statements carefully, try to understand it, ask questions, start solving it in the proper manner. Hand written – in my office – 28th of May

Question 1:

Write a function that has an array of size **N (character values – Capital), having 5** different keys. You have to find all the keys from the array in 1 traversal, after that you have to sort the key values in the given array in the ascending order in one traversal, on the basis of found key values form the array. **[We have solved a problem of three key values in the class, you have to modify that code].** You have, 1 to 5 different key values in array, and you have to decide which key value comes first and which comes after. **O(N) + O(N)**

Question 2:

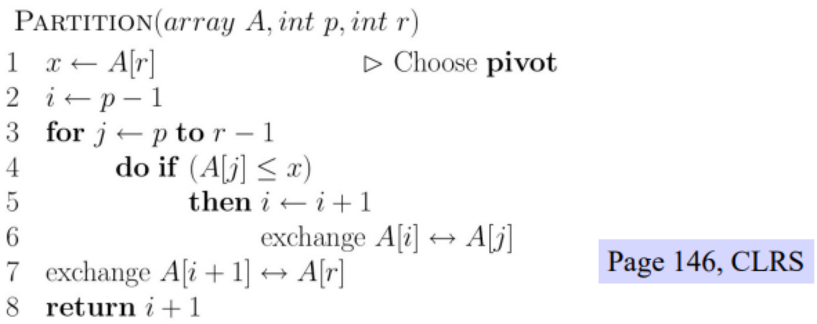
You have complete pseudo code for quick sort algorithm, Update **Partition** function used in the quicksort algorithm, that it will make some changes and place all smaller than pivot value on the left, all greater than pivot value on the right and all same as pivot value in the center, also keep in mind that what will be the updated quicksort calls after that function. You have to think that may be now function returns more than one value back.



QuickSort

```
QUICKSORT(array A, int p, int r)
if (p < r)
    then q ← PARTITION(A, p, r)
         QUICKSORT(A, p, q − 1)
         QUICKSORT(A, q + 1, r)
```

To sort array call QUICKSORT(A, 1, length[A]).



```
PARTITION(array A, int p, int r)
x ← A[r]                          ▷ Choose pivot
i ← p − 1
for j ← p to r − 1
    do if (A[j] ≤ x)
        then i ← i + 1
            exchange A[i] ↔ A[j]
exchange A[i + 1] ↔ A[r]
return i + 1
```

Page 146, CLRS

3	12	8	17	5	4	13	9	2	9	2	15	21	7	9

After 1 updated partition call

3	8	5	4	2	2	7	9	9	9	12	15	21	17	13

What should be our return values after this partition, and updated next 2 quicksort calls.

Question 3:

Write a function that has an array of size **N (character values – Capital), having 7** different keys (A, B, C, D, E, F, G). You have to sort the key values in the given array in the ascending order, in minimum number of updated Partition function calls. How many calls you will need, what update your solution will need in the updated partition function call to solve your problem here. We are assuming that array has all key value occurrences. Inside your function call all the work should be completed and you should design an updated partition function for this problem solution also.