***Quick Sort***

Quick Sort works in two parts one is making partition and the second one is performing quick sort function.

Here the main role is played by the pivot element which can be taken anything but I personally prefer the starting element of the array.

So pivot=start

The base condition will be when the high <= low it will be less when the pivot is low

And if it is the only element we do not want to sort only one element hence we will return it

Next the function of quick sort

Here we need to mention the base case first which I have described above

Then the next step is calling partition.

Partition function is used for getting the pivot element to it’s original or intended position this can be done by counting the number of elements that are smaller or equal to the pivot element which is also the first or starting element in our case so even if it is equivalent to the pivot element the counter will increase as we will check in if condition tha if the the element i is less than or equal to the pivot increase the count variable.

The next part after getting to know how many smaller or equal elements are there we need to change the pivot accordingly. So the new pivot will be equivalent to start which is the starting point of the given array to the partition function and not 0 because it will point to the first element of the main array.

Now I have to switch the element of starting index and pivot so that pivot reaches it’s destination and all the smaller elements are on the left whereas all the larger elements are on the right.

Next we need to create two variables namely I and j where I will point to the starting index and the j will point to the ending index of the partition function and not the whole array. These are taken in order to swap to their intended positions.

So we need to start a while loop where the condition will be

I<pivot and j>pivot

If the arr[i] > arr[pivot] and arr[j]>arr[pivot] this means that the ith element and jth element are both greater than the pivot element in this case we want to decrease the jth index because the ith element is not in position and we want to find which is smaller to pivot in order to put it in place of the ith element.

Next we need to check if arr[i]<= arr[pivot] this means that the ith element is equivalent or less than the pivot element we need these type of elements on the left so we will just increase the ith element by 1 because the element is perfectly in place.

Next we have the else part where we will check if the ith element is greater than the pivot element and jth element is smaller than the pivot element in this case we need to swap the ith element with the jth element after doing the swap the ith element will be perfectly in place as it was jth element.

And at last we will return the pivot

Now coming to the quick sort function again after partition it will call quick sort function for two parts one which is left part of the pivot and the right part of the pivot .