①

FDS lab

Modern College of Engineering

Shivajinagar, Pune 5. Name-Mahesh Jagtap

Roll No: 21027

Assignment no.B2

Title:- Quick sort Algorithm:

Problem statement:

Write a python program to store 1st year percentage of students in array. Write function for sorting array of floating point numbers in ascending order using quick sort & display top five scores.

Prerequisites:-

concept of Arrays, basics of sorting techniques, Quick sort Algorithm.

learning objectives:-

To implement & learn program for quick sort for percentage of students in ascending order.

Theory:

Quick sort, as the name suggests, sorts any list Very quickly. Quick sort is not stable search, but it is very fast & requires very less additional space.

It works on divide & conquer rule.

It divides input array in two partitions (recursively) & performs in-place sorting while doing so.

A ~~a~~ seperate partition() function is used for performing this in-place sorting at every iteration.

Working :

There are 2 phases in the Quick sort Algo.

① Division phase -
~~Did~~ Divide the array (list) into 2 halves by finding the pivot point to perform the partition of the array.
The In-place sorting happens in this partition process itself.

② Recursion phase -
- Call Quick sort on the left partition
- Call Quick sort on the right partition.

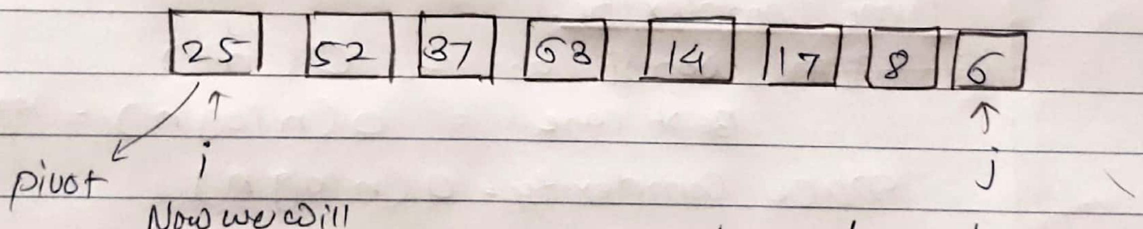
Quick Sort Algo divides the list into 3 main parts:

① Elements less than pivot element
② Pivot element
③ elements greater than pivot element

In the list of elements in below example, we have taken 25 as pivot. you can choose any element as pivot. (generally 1st, last or middle element is selected as pivot) so, after 1st pass, the list will be changed like this,

6 8 17 14 <u>25</u> 63 37 52

Hence after 1st pass, pivot will be set at its position, with all elements smaller to it on its left & all elements larger than it on the right.
Now, 6 8 17 14 & 63 37 52 are considered as two seperate lists, & same logic is applied on them, & we keep doing this until the complete list is sorted.



25	52	37	63	14	17	8	6

Now we will keep on traversing the list if a[i]< pivot & a[i]!= pivot

here also we keep on traversing the list from back, if a[j]> pivot & a[j]!= pivot

If both sides we find the element not satisfying their respective conditions we swap them And keep reapeating this.

Quick sort pivot Algorithm:-

step 1: Choose the highest index value as pivot
step 2: Take two variables to point left & right of this list excluding pivot.
step 3: left points to the low index.
step 4: right points to the high index.
Step 5: while value at left is less than pivot move right
Step 6: while value at right is greater than pivot move left.
Step 7: If both steps 5 & step 6 doesnot match swap left & right.
Step 8: If left $\geq$ right, the point where they met is new pivot

Time Complexity:

~~Conclusion~~: worst case = $O(n^2)$
Best case = $O(n \log n)$ = Avg.
Space Complexity = $O(n \log n)$

Conclusion:

In this way, we have studied Quick Sort in detail & implemented it to solve given problem.