15CSE374
INTRODUCTION TO DATA STRUCTURES AND ALGORITHMS

Sarath tv

Last lecture

- Sorting
- Insertion sort
- Selection sort
- Heap sort

Divide-and-Conquer

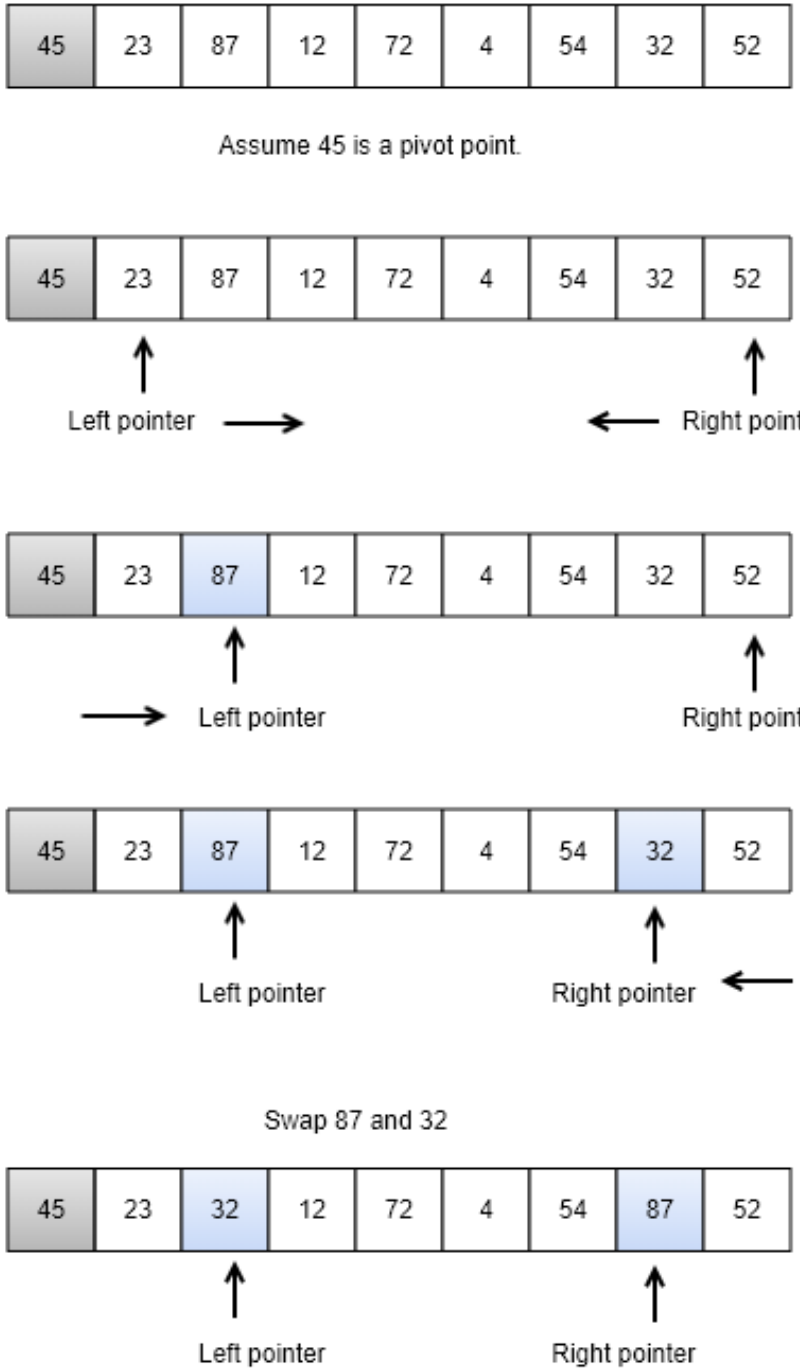
- Use recursion in an algorithmic design pattern.

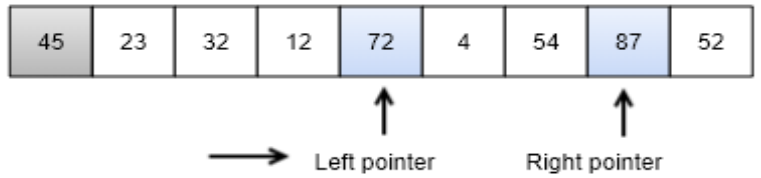
1. ***Divide:*** If the input size is smaller than a certain threshold (say, one or two elements), solve the problem directly using a straightforward method and return the solution so obtained. Otherwise, divide the input data into two or more disjoint subsets.
2. ***Conquer:*** Recursively solve the subproblems associated with the subsets.
3. ***Combine:*** Take the solutions to the subproblems and merge them into a solution to the original problem.

Quick sort

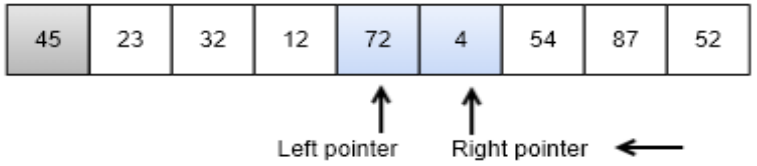
- Falls under the divide and conquer class of algorithms,
- We break (divide) a problem into smaller chunks that are much simpler to solve (conquer).
- Partitioning a given list
- We first select a pivot. All the elements in the list will be compared with this pivot. At the end of the partitioning process, all elements that are less than the pivot will be to the left of the pivot, while all elements greater than the pivot will lie to the right of the pivot in the array.

Example


45 23 87 12 72 4 54 32 52
Assume 45 is a pivot point.
45 23 87 12 72 4 54 32 52
Left pointer
Right pointer
45 23 87 12 72 4 54 32 52
23<45, continue moving to the right
87<45, stop here
Left pointer
Right pointer
45 23 87 12 72 4 54 32 52
52>45, continue moving to the left
32<45, stop here
Left pointer
Right pointer
Swap 87 and 32
45 23 32 12 72 4 54 87 52
Left pointer
Right pointer

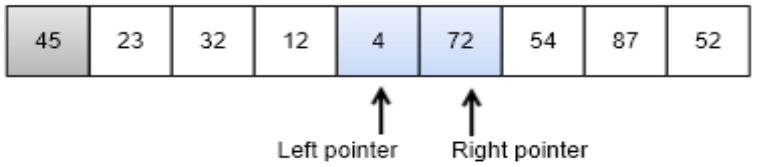


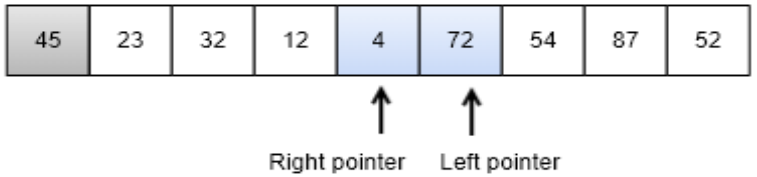
12<45, continue moving to the right
72>45, stop here

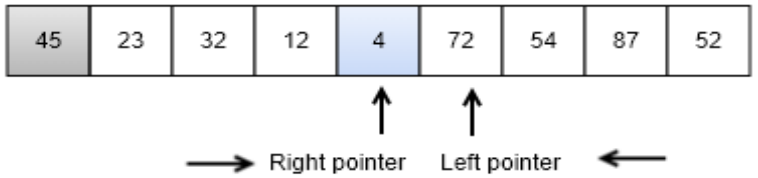


54>45, continue moving to the left
4<45, stop here

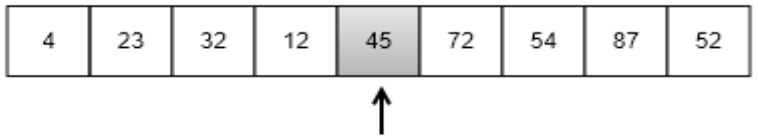
Swap 72 and 4

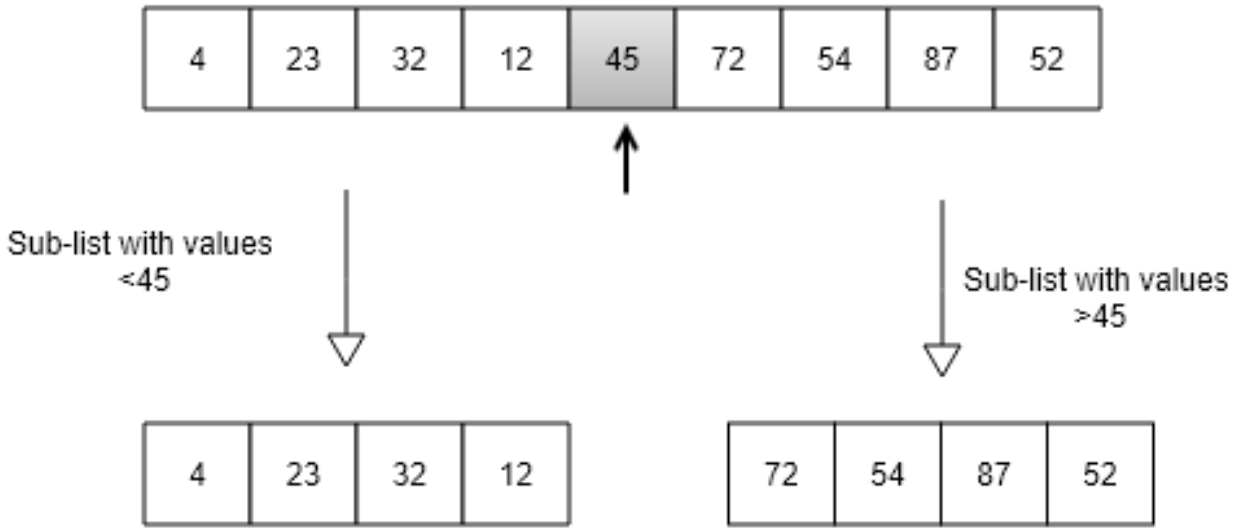






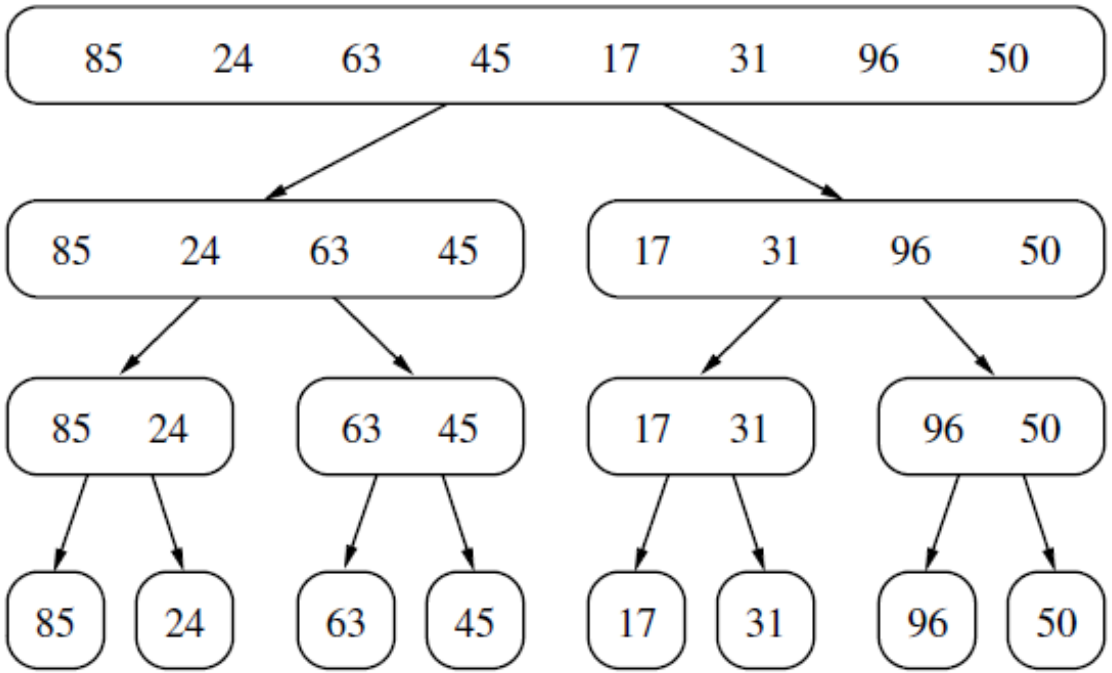
72>45, stop here
4<45, stop here
Swap 45 and 4.

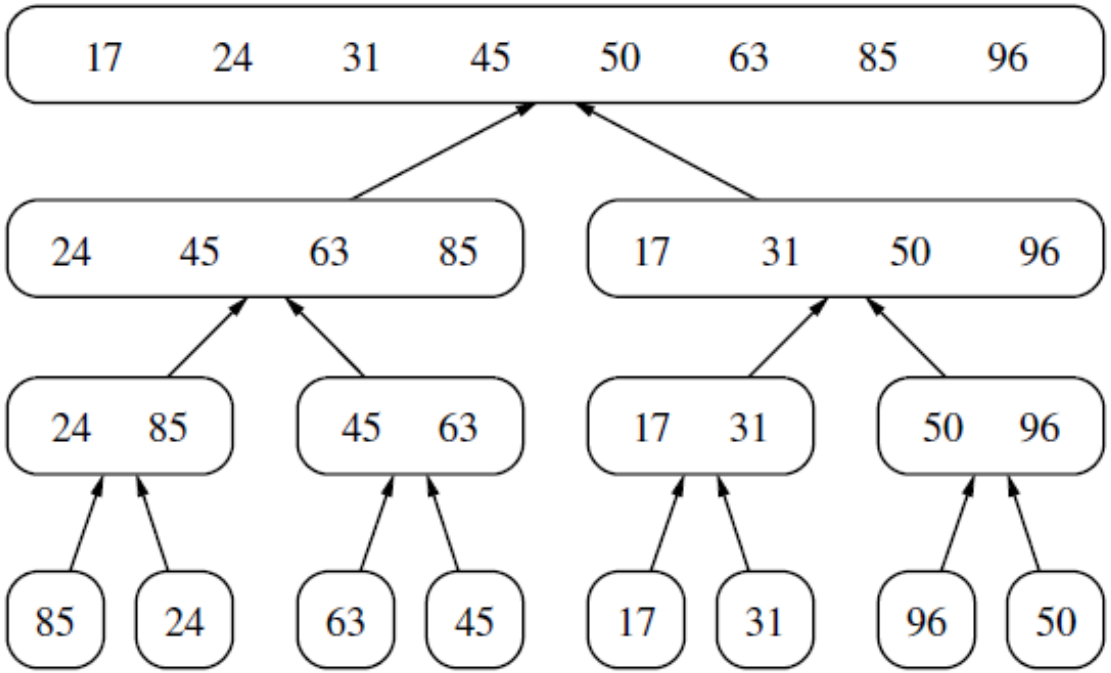


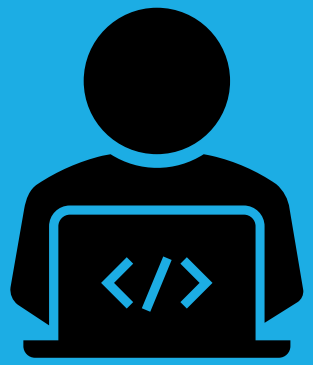

4
23
32
12
45
72
54
87
52
Sub-list with values <45
Sub-list with values >45
4
23
32
12
72
54
87
52

Merge-Sort

- Visualize an execution of the merge-sort algorithm by means of a binary tree T, called the ***merge-sort tree***.
- Each node of t represents a recursive invocation
- Splits a list in half, and keeps splitting the list by 2 until it only has singular elements
- We recursively split the list in half until we have lists with size one. We then merge each half that was split, sorting them in the process. Sorting is done by comparing the smallest elements of each half. The first element of each list are the first to be compared. If the first half begins with a smaller value, then we add that to the sorted list. We then compare the second smallest value of the first half with the first smallest value of the second half.
- Every time we select the smaller value at the beginning of a half, we move the index of which item needs to be compared by one.


85 24 63 45 17 31 96 50
85 24 63 45
17 31 96 50
85 24
63 45
17 31
96 50
85
24
63
45
17
31
96
50


17 24 31 45 50 63 85 96
24 45 63 85
17 31 50 96
24 85
45 63
17 31
50 96
85
24
63
45
17
31
96
50



THANK YOU!!!!!