Department of
Computer Science and Engineering

Title: Implement Quick Sort Algorithm

Algorithms Lab
CSE 206

Green University of Bangladesh

1 Objective(s)

- To understand how Merge Sort algorithm works.
- To implement Merge Sort algorithm in Java.

2 Problem analysis

Quick Sort is a type of divide and conquer algorithm for sorting an array, based on a partitioning. It works by selecting a 'pivot' element from the array and partitioning the other elements into two sub-arrays, according to whether they are less than or greater than the pivot. Pivot element can be any element from the array, it can be the first element, the last element or any random element. We will consider the rightmost element or the last element as the pivot. This algorithm works by dividing the list into three main parts:

i Elements less than the Pivot element.

ii Pivot element(Central element).

iii Elements greater than the pivot element.

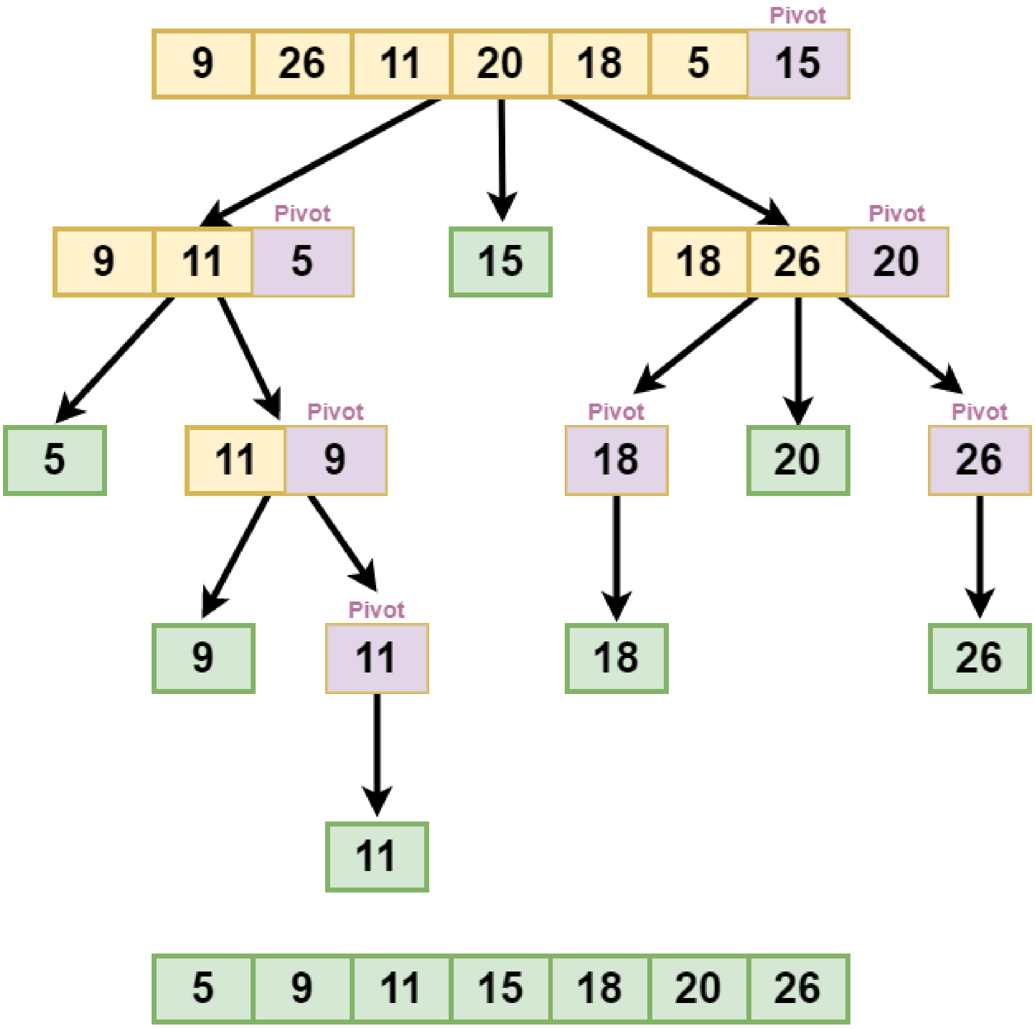

Figure 1: Quick-Sort illustrated

3 Algorithm

Algorithm 1: Quick Sort Algorithm

Input: An array A, index lo and hi defining range
Output: Sorted array A within index range lo and hi

```
if $lo < hi$ then
    $p \leftarrow partition(A,\ lo,\ hi)$
    $quicksort(A,\ lo,\ p-1)$
    $quicksort(A,\ p+1,\ hi)$
end
```

Algorithm 2: Partition Algorithm

Input: An array A, index lo and hi defining range
Output: A pivot p, with all low element on left and height element on right

```
$pivot \leftarrow A[\lfloor \frac{hi+lo}{2} \rfloor]$
$i \leftarrow lo - 1$
$j \leftarrow hi + 1$
for $forever$ do
    $i \leftarrow i + 1$
    while $A[i] < pivot$ do
        $j \leftarrow j - 1$
    end
    while $A[j] > pivot$ do
        if $i \geq j$ then
            return $j$
        end
    end
    swap $A[i]$ with $A[j]$
end
```

4 Implementation in Java

```java
package quicksort;

public class QuickSort {
    public static void main(String[] args) {
        int i;
        int[] arr = {5, 9, 11, 15, 18, 20, 26};
        quickSort(arr, 0, arr.length-1);
        System.out.println("The sorted array is: ");
        for (i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
    }

    public static int partition(int a[], int beg, int end) {
        int left, right, temp, loc, flag;
        loc = left = beg;
        right = end;
        flag = 0;
        while (flag != 1) {
            while ((a[loc] <= a[right]) && (loc != right)) {
                right--;
            }
            if (loc == right) {
                flag = 1;
            } else if (a[loc] > a[right]) {
                temp = a[loc];
                a[loc] = a[right];
                a[right] = temp;
                loc = right;
            }
            if (flag != 1) {
                while ((a[loc] >= a[left]) && (loc != left)) {
                    left++;
                }
                if (loc == left) {
                    flag = 1;
                } else if (a[loc] < a[left]) {
                    temp = a[loc];
                    a[loc] = a[left];
                    a[left] = temp;
                    loc = left;
                }
            }
        }
        return loc;
    }

    static void quickSort(int a[], int beg, int end) {
        int loc;
        if (beg < end) {
            loc = partition(a, beg, end);
            quickSort(a, beg, loc - 1);
            quickSort(a, loc + 1, end);
        }
    }
}
```

5 Sample Input/Output (Compilation, Debugging & Testing)

Output:

```
The sorted array is:
5
9
11
15
18
20
26
```

6 Discussion & Conclusion

Based on the focused objective(s) to understand about the divide & conquer quick sort, the additional lab exercise made me more confident towards the fulfilment of the objectives(s).

7 Lab Task (Please implement yourself and show the output to the instructor)

1. Draw the step-by-step solution of Quick Sort for the following array.

20	7	15	9	35	4	1	11	7	16

2. Implement and obtain the output of Quick Sort for the same array.

8 Lab Exercise (Submit as a report)

- Implement Quick sort on a linked list.

9 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.