1 Sorting: Mechanical Practice

Show the steps taken by each sort on the following unordered list:

106, 351, 214, 873, 615, 172, 333, 564

(a) Quicksort. After each partition during the algorithm, write the ordering of the list, circle the pivot that was used for that partition, and box the sub-array being partitioned. Assume that the pivot is always the first item in the sublist being sorted and that the array is sorted in place.

(b) Merge sort. Show the intermediate merging steps.

(c) LSD radix sort. Show the ordering of the list after each round of counting sort.
2 Sorting: Identification

Match the sorting algorithms to the sequences, each of which represents several intermediate steps in the sorting of an array of integers. Assume that for quicksort, the pivot is always the first item in the sublist being sorted.

Algorithms: Quicksort, merge sort, heapsort, MSD radix sort, insertion sort.

(a) 12, 7, 8, 4, 10, 2, 5, 34, 14
    7, 8, 4, 10, 2, 5, 12, 34, 14
    4, 2, 5, 7, 8, 10, 12, 14, 34

(b) 23, 45, 12, 4, 65, 34, 20, 43
    4, 12, 23, 45, 65, 34, 20, 43

(c) 12, 32, 14, 11, 17, 38, 23, 34
    12, 14, 11, 17, 23, 32, 38, 34

(d) 45, 23, 1, 65, 34, 3, 76, 25
    23, 45, 1, 65, 3, 34, 25, 76
    1, 23, 45, 65, 3, 25, 34, 76

(e) 23, 44, 12, 11, 54, 33, 1, 41
    54, 44, 33, 41, 23, 12, 1, 11
    44, 41, 33, 11, 23, 12, 1, 54

3 Runtimes, Part 1: Comparison Sorts

Fill in the best and worst case runtimes of the following comparison sorting algorithms with respect to \(N\), the length of the list being sorted.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Worst case</th>
<th>Best case</th>
<th>Stable? (Yes/No)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection sort</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Insertion sort</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Merge sort</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quicksort</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heapsort</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4 Runtimes, Part 2: Counting Sorts

Fill in the best and worst case runtimes of the following counting sorting algorithms with respect to $N$, the length of the list being sorted. Assume we are sorting integers and $L$ is the average number of digits in the integers being sorted.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Worst case</th>
<th>Best case</th>
<th>Stable? (Yes/No)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distribution counting</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSD radix sort</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSD radix sort</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5 Comparing Algorithms

(a) Sometimes insertion sort can be more efficient than merge sort. Give an example of an input array that demonstrates this.

(b) When might you decide to use radix sort over a comparison sort, and vice versa?