CSE 214 Spring 2023 (Student Version)

Recitation 11 - Sorting

1. [5 minutes] Table of Time Complexities

Fill in the following table giving the best/worst case order of complexity for the following sorting algorithms. Briefly describe when these cases occur.

Sort Algorithm	Best Case	Worst Case
Bubble	$O(n)$ Arr already sorted. Break.	$O(n^2)$
Selection	$O(n^2)$	$O(n^2)$
Insertion	$O(n)$ Alr sorted	$O(n^2)$
Quick	$O(n \log n)$ lucky, middle partition	$O(n^2)$ Unlucky pivot selection.
Merge	$O(n \log n)$	$O(n \log n)$
Heap	$O(n \log n)$	$O(n \log n)$

2. [5 minutes] Bubble Sort:
This is probably the simplest way to sort an array of objects. The basic idea is to compare two neighboring objects, and to swap them if they are in the wrong order. Sort the following sequence of numbers in an ascending order(stored in an array)by using bubble sort.

54 26 93 17 77 31 44 55 20

26 54 17 77 31 44 55 20 93

26 17 54 31 44 55 20 77 93

17 26 31 44 54 20 55 77 93

17 26 31 44 20 54 55 77 93

17 26 31 20 44 54 55 77 93

17 26 20 31 44 54 55 77 93

17 20 26 31 44 54 55 77 93

3. [5 minutes] Trace how Selection Sort sort run on the following array of integers in an ascending order, showing the results after each run of the outer loop.

{9, 96, 10, 62, 98, 7, 53, 1, -3}

-3 96 10 62 98 7 53 1 9

-3 1 10 62 98 7 53 96 9

-3 1 7 62 98 10 53 96 9

-3 1 7 9 98 10 53 96 62

-3 1 7 9 10 98 53 96 62

-3 1 7 9 10 53 98 96 62

-3 1 7 9 10 53 62 96 98

-3 1 7 9 10 53 62 96 98

-3 1 7 9 10 53 62 96 98

4. **[5 minutes] Given the list of numbers { 14, 3, 7, 11, 9, 8, 13, 4, 2, 12}, give the trace for Insertion Sort (ascending order).**

14 | 3 7 11 9 8 13 4 2 12

3 14 | 7 11 9 8 13 4 2 12

3 7 14 | 11 9 8 13 4 2 12

3 7 11 14 | 9 8 13 4 2 12

3 7 9 11 14 | 8 13 4 2 12

3 7 8 9 11 14 | 13 4 2 12

3 7 8 9 11 13 14 | 4 2 12

3 4 7 8 9 11 13 14 | 2 12

2 3 4 7 8 9 11 13 14 | 12

$\boxed{2\ 3\ 4\ 7\ 8\ 9\ 11\ 12\ 13\ 14}$

5. **[10 minutes] Merge Sort and Quick Sort**

Merge Sort and Quick Sort are divide-and-conquer algorithms. This means rather than attack the whole list at once like selection sort and insertion sort, they break the list up into smaller, easier to deal with pieces and sort those until the whole list is sorted. In the case of merge sort, lists are broken down until they are size 1. Two sorted lists of size n can be merged into a single sorted list of size 2n using the merge algorithm. Quick Sort works by splitting the list in half and Quick Sorting each half. Remember that a list of one element is sorted by definition.

Run quicksort and Merge Sort on the following list {7, 2, 4, 1, 8, 6, 9, 5}:

Quick Sort:

7 2 4 1 8 6 9 5

largest# <5

2 7 4 1 8 6 9 | 5

2 4 7 1 8 6 9 | 5

2 4 1 7 8 6 9 | 5

$\boxed{2\ 4\ 1}$ 5 $\boxed{8\ 6\ 9\ 7}$

$\boxed{2\ 4 \mid 1}$ 5 $\boxed{8\ 6\ 9 \mid 7}$

1 $\boxed{4 \mid 2}$ 5 $\boxed{6}$ 7 $\boxed{8\ 9}$

$\boxed{1\ 2\ 4\ 5\ 6\ 7\ 8\ 9}$

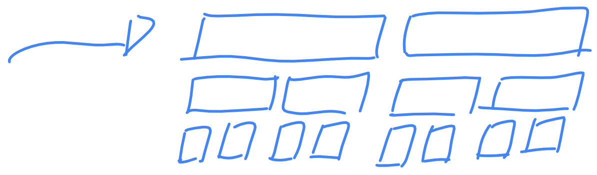
Merge Sort: 7 2 4 1 8 6 9 5

6 7 2 4 1 8 6 9 5

27 14 68 59

1247 5689

12456789



6. **[15 minutes]** Simulate heapsort to sort the array [29, 30, 2, 14, 49, 12, 4, 8] in ascending order.

49 30 12 14 29 2 4 8

30 29 12 14 8 2 4 | 49

29 14 12 4 8 2 | 30 49

14 8 12 4 2 | 29 30 49

12