EECS 233 SI Session
Leader: Bertram Su
March 27, 2019

Objectives:
Upon completion of this SI session, participants will be able to:

1. Recognize the big O expression of selection and insertion sort
2. Determine how arrays will look after a few cycles of selection and insertion sort

Foundations:

1. What are the steps for selection sort? Words: decrease, swap, increase, smallest
   1. find the Smallest entry
   2. swap the smallest entry with the first unsorted entry
   3. increase your sorted range / decrease your unsorted range

2. What are the steps for insertion sort? Words: sorted, increase, decrease, shift
   1. Compare the first unsorted element with the sorted elements
   2. while searching for place to insert, shift then insert
   3. increase your sorted range decrease your unsorted range

Exercises:

3. Draw the phases of the array after each swap when we use selection sort

0	1	2	3	4
22	**44**	**34**	**21**	**6**

→

0	1	2	3	4
6	44	34	21	22

→

0	1	2	3	4
6	21	34	44	22

0	1	2	3	4
6	21	22	44	34

→

0	1	2	3	4
6	21	22	34	44

4. Draw the phases of the array after each insertion when we use insertion sort

0	1	2	3	4
22	**44**	**34**	**21**	**6**

→

22	34	44	21	6

21	22	34	44	6

→

6	21	22	34	44

Available methods:

swap(int[] arr, int index1, int index2) index

findMin(int[] arr, int index) //finds min ✓of an array from [index1, array.size)

5. Using the available methods, write the following code.

```
public void selectionSort(int[] arr){
      for(i=0; i< arr.size; i++){
        int min = find min (arr, i);
        swap (i, min)

}
```

What's the Big O of this? n^2

6. Using the following is code for insertion sort from lecture.

```
static void insertionSort(int[] arr) {
        for (int i = 1; i < arr.length; i++) {
            int toInsert = arr[i];
            for (j = i; j > 0 && toInsert < arr[j-1]; j--) //Loop X
                arr[j] = arr[j-1];
            arr[j] = toInsert;
        }
}
```

What's the Big O of this? n^2

What two things is loop X doing?

finding the place to insert and shifting

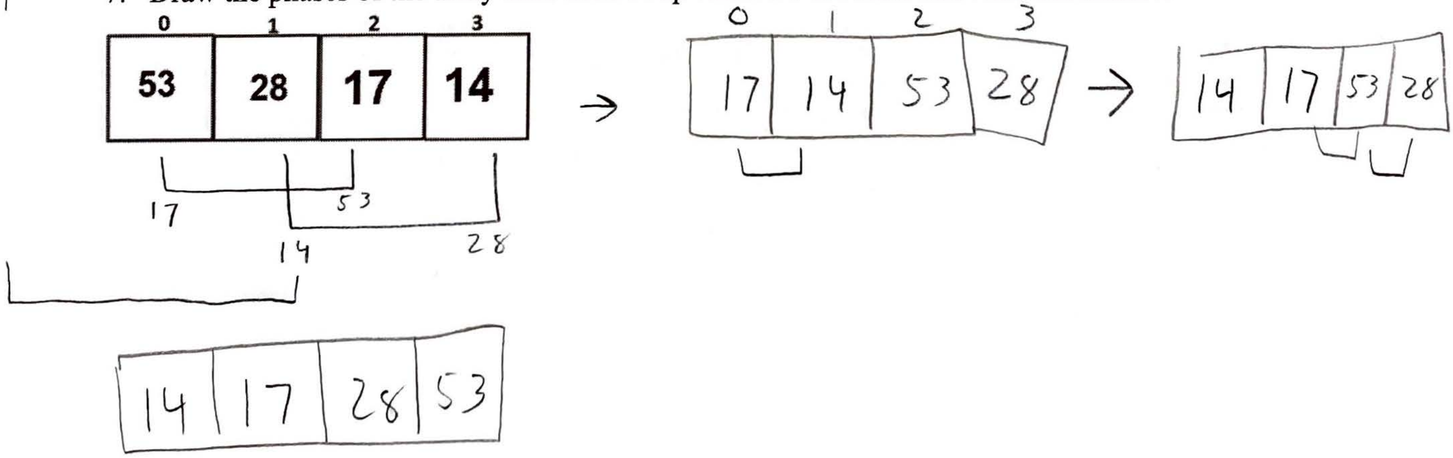
Dr. Ayday talked about an optimization. What was it, where would it go?

Loop x

Binary search

-1

7. Draw the phases of the array after each swap when we use shell sort with increment 2



Summary

8. Using insertion sort, sort the following array and draw the array after each insertion. What do you notice about the runtime?

0	1	2	3
0	7	17	23

$O(n)$

9. What is the best case and worst case runtime for selection and insertion sort?

Best : $O(n^2)$ SS $O(n)$ IS

Worst : $O(n^2)$ $O(n^2)$

10. Every now and then a random order number will be sent to you for storage. Each time an integer comes in you want to add it to your array and sort them. Would we want to use selection or insertion sort?

IS

11. [Hard] When would we use selection sort? What makes insertion sort expensive?

when shifting is expensive

Upcoming Events and Suggestions for Further Study:

Events:

- Next SI session is Sunday from 1:00 to 2:30 at Sears 336

Further Study

- https://www.geeksforgeeks.org/selection-sort/
  - It has a great visualization, explanation, and code of selection sort
- https://www.geeksforgeeks.org/insertion-sort/
  - same features as above but for insertion sort

EECS 233 SI Session
Leader: Bertram Su
November 10, 2019

3	2	1

2	3	1

2	1	3

Objectives:

Upon completion of this SI session, participants will be able to:

1. Recognize the big O expression of bubble and quick sort
2. Determine how arrays will look after cycles of bubble and quick sort

Foundations:

1. What are the steps for quick sort? Words: right, pivot, left
   1) Choose a pivot
   2) move elements to the left if < pivot and elements to the right if > pivot
   3) perform 1 and 2 again on both subarrays until sorted

2. What are the steps for bubble sort? Words: swap, order
   1) From left to right, swap adjacent elements if they are out of order
   2) Repeat 1 until sorted

Exercises:

Using quick sort, draw the partition phases for this array. pivot = arr[(first + last)/2]

0	1	2	3
68 X	13 X	37 ✓	74 ✓

Pivot = 13

→

0	1	2	3
13	68 x	37 x	74 ✓

Pivot = 37

0	1	2	3
13	37	68 x	74

Pivot = 68

3. Using bubble sort, draw the phases of the array after an element gets completely bubbled up.

0	1	2	3
1	32	23	3

→

1	23	3	32

1	3	23	32

4. The following is code straight from lecture.

```
static void bubbleSort(int[] arr, int n) {
        for (int i = n – 1; i > 0; i--) {
                for (int j = 0; j < i; j++) {
                        if (arr[j] > arr[j+1])
                            swap(arr, j, j+1);
                }
        }
}
```

What is the Big O runtime of this? What is the best case runtime of this?

Big O (n^2) Best n^2

If it's even possible, describe in words how we would optimize the best case.

if no swaps, break /return

5. Using bubble sort, draw the phases of the array after an element gets completely bubbled up.

0	1	2	3	4
67	1	8	2	32

→

1	8	2	32	67

1	2	8	32	67

Using quick sort, draw the partition phases for this array. pivot = arr[(first + last)/2]

0	1	2	3
26	7	28	3

Pivot = 7

3	7	28	26

3	7

26	28

6. The following code is from our lecture notes. Fill in the blanks

```
static int partition(int[] arr, int first, int last){
        int pivot = arr[(first + last)/2];
        int i = first - 1;// index going from left to right
        int j = last + 1;// index going from right to left
        while (true) {
        do {
                __i__++;
        } while (arr[i] < __pivot__);
        do {
                j--;
        } while (__arr[j] > pivot__);
                if (i < j)
                  swap(arr, i, j);
                else
                  return j;// arr[j] = end of left array
        }
}
```

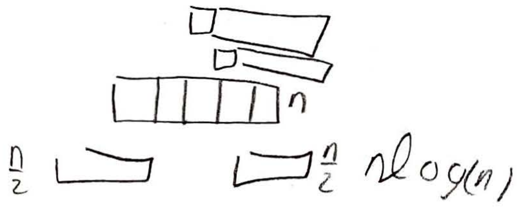
Summary

7. What is the best case and worst case runtime of the lecture's bubble sort?

O(n²)

8. What is the best case and worst case of quick sort?

best: n·log n   Worst: n²



9. [Hard] Given the following conditions, what sorting method(s) would you use?
    a. The data is mostly sorted, but there are a few out of place elements
    b. The data is completely backwards from being sorted.

a) insertion

b) quicksort

Upcoming Events and Suggestions for Further Study:

Events:

- Next SI session is Thursday from 6:00 to 7:30 at Sears 336

Further Study:

- https://www.geeksforgeeks.org/quick-sort/
    - It has a great visualization, explanation, and code of quick sort
- https://www.geeksforgeeks.org/bubble-sort/
    - Same but bubble sort