# CSE 12 – Basic Data Structures and Object-Oriented Design
# Lecture 11

Greg Miranda, Spring 2021

This lecture is being recorded

# Announcements

- Quiz 11 due Monday @ 12pm
- PA3 due tonight @ 11:59pm (open collaboration)
- Survey 4 due Friday @ 11:59pm
- Exam 1 on Friday (no class)
  - Released @ 2pm on Friday
  - Closes @ 6pm on Saturday
  - More details to be released on Piazza soon
    - Lectures 1 – 8
    - Up to and including PA3
    - 90 minutes
    - No make-ups

28 hour

multiple choice
coding problems
short answers

# Topics

- Questions on Lecture 11?
- Big Theta
- Sorting

Count how many times each line executes, then say which Θ( ) **statement(s) is(are) true**.

```
int maxDifference(int[] arr){
   max = 0;
   for (int i=0; i<arr.length;  i++) {
      for (int j=0; j<arr.length; j++) {
         if (arr[i] – arr[j] > max)
            max = arr[i] – arr[j];
      }
   }
   return max;
}
```

| Line # |
|---|
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |

A. $f(n) = \theta(2^n)$
B. $f(n) = \theta(n^2)$
C. $f(n) = \theta(n)$
D. $f(n) = \theta(n^3)$
E. Other/none/more

*(assume n = arr.length)*

Count how many times each line executes, then say which Θ( ) **statement(s) is(are) true**.

```
int sumTheMiddle(int[] arr){
   int range = 100;
   int start = arr.length/2 – range/2;
   int sum = 0;
   for (int i=start; i<start+range;  i++)
   {
      sum += arr[i];
   }
   return max;
}
```

A. $f(n) = \theta(2^n)$
B. $f(n) = \theta(n^2)$
C. $f(n) = \theta(n)$
D. $f(n) = \theta(1)$
E. None of these

*(assume n = arr.length)*

# Selection Sort – what does it print out?

```
import java.util.Arrays;
public class Sort {
public static void sortA(int[] arr) {
  for(int i = 0; i < arr.length; i += 1) {
    System.out.print(Arrays.toString(arr) + " -> ");
    int minIndex = i;
    for(int j = i; j < arr.length; j += 1) {
      if(arr[minIndex] > arr[j]) { minIndex = j; }
    }
    int temp = arr[i];
    arr[i] = arr[minIndex];
    arr[minIndex] = temp;
    System.out.println(Arrays.toString(arr));
  }
}
}
```

```
Sort.sortA(new int[]{ 53, 83, 15, 45, 49 });
[53, 83, 15, 45, 49] ->
```

# Insertion Sort – what does it print out?

```
import java.util.Arrays;
public class Sort {
public static void sortB(int[] arr) {
  for(int i = 0; i < arr.length; i += 1) {
    System.out.print(Arrays.toString(arr) + " -> ");
    for(int j = i; j > 0; j -= 1) {
      if(arr[j] < arr[j-1]) {
        int temp = arr[j-1];
        arr[j-1] = arr[j];
        arr[j] = temp;
      }
    }
    System.out.println(Arrays.toString(arr));
  }
}
}
```

```
Sort.sortB(new int[]{53, 83, 15, 45, 49 });
[53, 83, 15, 45, 49] ->
```