CSCI 136
Data Structures & Advanced Programming
(Lecture 11)

Sorting 2

Jon Park

Mar 1, 2017

Announcements

- Lab 4
  - You may work with a partner (But be sure to understand how the code works!)

Last Time

- Sorting
  - Selection Sort
  - Bubble Sort

Today's Outline

- Review
  - Selection Sort
  - Bubble Sort
- Sorting (Continued)
  - Insertion Sort
  - Merge Sort

Selection Sort

```
public static void selectionSort(int[] data) {
    for (int curN = data.length-1; curN > 0; curN--){
        int maxIdx = 0;
        for (int i = 1; i <= curN; i++){
            if(data[i] > data[maxIdx])
                maxIdx = i;
        }
        swap(data, maxIdx, curN);
    }
}
```

put maxIdx

Bubble Sort

```
public static void bubbleSort(int[] data) {
    for (int currN = data.length-1; currN > 0; currN--){
        boolean swapped = false;
        for (int i = 1; i <= currN; i++){
            if (data[i-1] > data[i]){
                swap(data, i, i-1);
                swapped = true;
            }
        }
        if (!swapped)
            break;
    }
}
```

Best

$O(n^2)$

$O(n)$

Today's Outline

- Review
  - Selection Sort
  - Bubble Sort
- Sorting (Continued)
  - Insertion Sort
  - Merge Sort

Sorting a Deck of Cards

Time Complexity:

A. O(n) ← best

B. O(n log n)

C. $O(n^2)$ ← worst, Ave

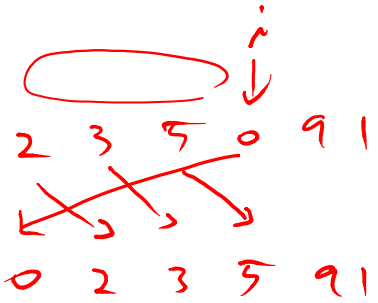
D. $O(n^3)$

E. Not sure

Insertion Sort

- [5 7 0 3 4 2 6 1]
- [5 7 0 3 4 2 6 1]
- [0 5 7 3 4 2 6 1]
- [0 3 5 7 4 2 6 1]
- [0 3 4 5 7 2 6 1]
- [0 2 3 4 5 7 6 1]
- [0 2 3 4 5 6 7 1]
- [0 1 2 3 4 5 6 7]

Insertion Sort




```
public static void insertionSort(int[] data) {
    for (int i = 1; i <= data.length - 1; i++) {
        int temp = data[i];
        int j;
        for (j = i; j > 0; j--) {
            if (temp < data[j-1])
                data[j] = data[j-1];
            else
                break;
        }
        data[j] = temp;
    }
}
```

Insertion Sort Summary

- Overview
  - After *ith* iteration, at least *i* items are sorted.
  - During *ith* iteration, take the first item in the unsorted portion of the list and **insert** it to the "correct" location in the sorted portion.
- Time complexity:
  - Best case: O(n)
  - Worst case: $O(n^2)$
  - Average case: $O(n^2)$

Today's Outline

- Review
  - Selection Sort
  - Bubble Sort
- Sorting (Continued)
  - Insertion Sort
  - Merge Sort

Sorting a Deck of Cards

Time Complexity:

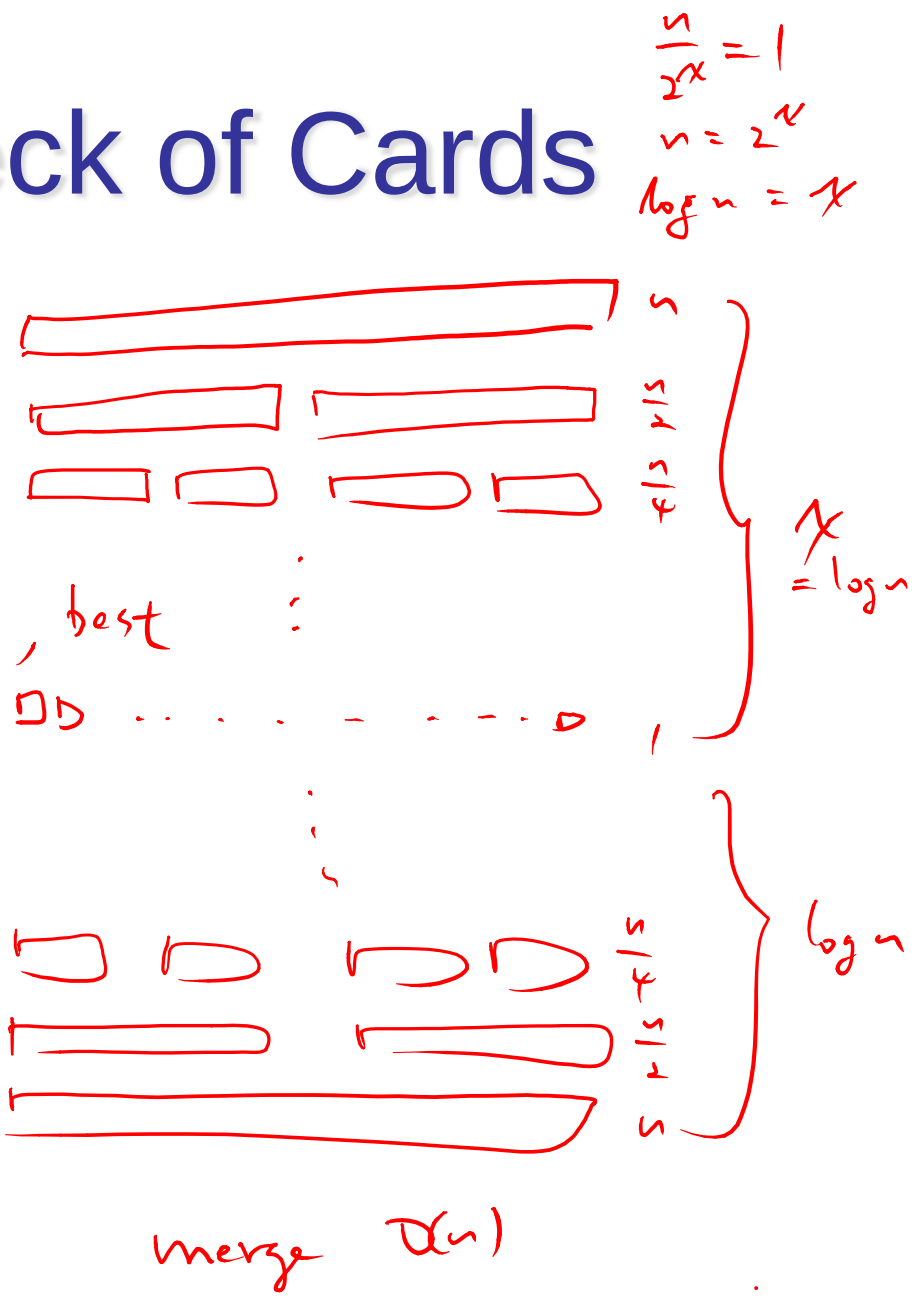
A. O(n)

B. O(n log n)

C. O(n^2)

D. O(n^3)

E. Not sure

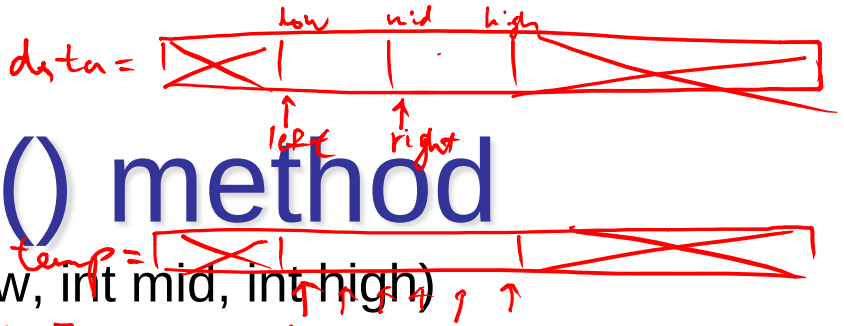


Merge Sort

- [8 14 29 1 17 39 16 9]
- [8 14 29 1] [17 39 16 9]
- [8 14] [29 1] [17 39] [16 9]
- [8] [14] [29] [1] [17] [39] [16] [9]
- [8 14] [1 29] [17 39] [9 16]
- [1 8 14 29] [9 16 17 39]
- [1 8 9 14 16 17 29 39]

split

merge



Aside: merge() method

public static void merge(int[ ] data, int low, int mid, int high)

```
int[] temp = new int[data.length];
int left = low;
int right = mid;
int cur = low;
while (left <= mid-1 && right <=high) {
    if( data[left] < data[right])
        temp[cur++]= data[left++];
    else
        temp[cur++]= data[right++];
}
```

//copy from temp to data