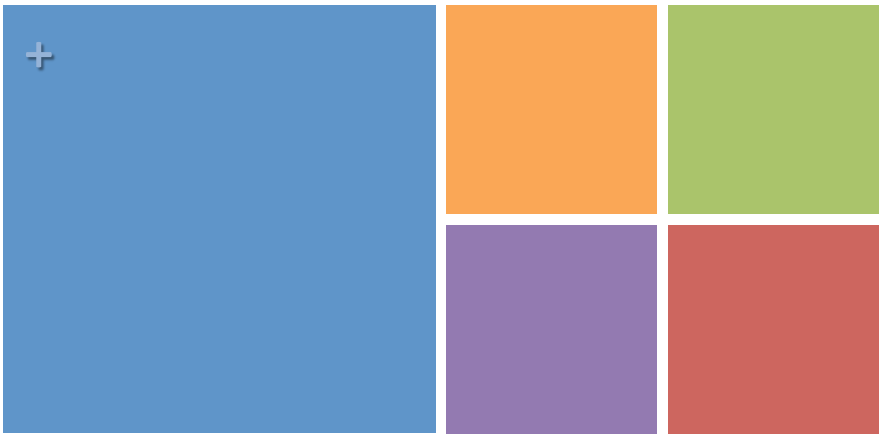

Lecture 7: Induction & Sorting

Today

- Reading
  - JS Ch. 5.2 – 5.3 (Recursion, Design), Ch. 6 (Sorting)
- Objectives
  - Induction
    - Correctness of Selection sort
    - Complexity of Selection sort

Announcements

- Assignment 2 starter code posted
  - How to copy the starter code to my workspace?
  - JavaDoc – what's expected?
- Quiz on Friday
  - Big-O
  - Induction
- Harvey Mudd Career Fair

Induction

- Let P(n) be some proposition
- To prove P(n) is true for all $n \geq 0$
  - (Step One) Base case: Prove P(0)
  - (Step Two) Assume P(k) is true for $k \geq 0$
  - (Step Three) Use this assumption to prove P(k+1).

Induction

- Practice Examples
  - Prove $1 + 2 + \ldots + n = [n(n+1)]/2$ for all $n \geq 0$
  - Prove $2^0 + 2^1 + \ldots + 2^n = 2^{n+1} - 1$ for all $n \geq 0$
  - Prove $2^n < n!$ for all $n \geq 4$
- Induction can be used to prove
  - Mathematical statements
  - Correctness of an algorithm
  - Complexity of an algorithm

Selection Sort

14	30	10	26	34	18	5	20
5	30	10	26	34	18	14	20
5	10	30	26	34	18	14	20
5	10	14	26	34	18	30	20
5	10	14	18	34	26	30	20

1. Find smallest
2. Swap
3. Repeat

Selection Sort

```
/**
 * Sorts an integer array using iterative selection sort
 * @param array array of integers to be sorted
 */
private static void selectionSortIterative(int[] array) {

     for(int i = 0; i < array.length; ++i) {
          int min = indexOfSmallest(array, i);
          swap(array, i, min);
     }
}
```

Selection Sort (helper)

```
/**
 * @param array array of integers
 * @param startIndex valid index into array
 * @return index of smallest value in array[startIndex...n]
 */
protected static int indexOfSmallest(int[] array, int startIndex) {

     int smallest = startIndex;
     for(int i = startIndex+1; i < array.length; ++i) {
          if(array[i] < array[smallest]) {
               smallest = i;
          }
     }
     return smallest;
}
```

Correctness of Selection Sort using Induction (on board)

- Consider what must be true after every iteration of the for-loop in selectionSortIterative

Complexity of Selection sort using Induction (on board)

- Count the number of comparisons performed for each iteration of the for-loop in selectionSortIterative

Strong Induction

- Sometimes need to assume more than just the previous case, so instead
  - Prove P(0)
  - For n > 0, use P(k) *for all* k < n as assumption in order to prove P(n).

Proof Example

- fastPower(x,n) algorithm to calculate x^n:
  - if n == 0 then return 1
  - if n is even, return fastPower(x*x,n/2)
  - if n is odd, return x*fastPower(x,n-1)
- Proof by induction on n (on board)
  - Base case: n ==0
  - Induction case: Assume fastPower(x,k) is x^k for all k < n. Show fastPower(x,n) is x^n