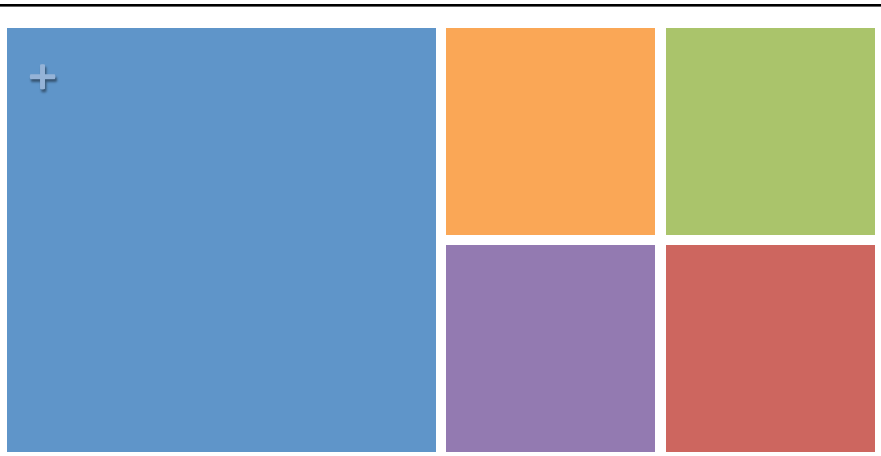

Lecture 7: Induction & Sorting

Today

- Reading
  - JS Ch. 5.2 – 5.3, Ch. 6
- Objectives
  - Induction
    - Correctness of Selection sort
    - Complexity of Selection sort
- Reminder
  - Quiz on Friday

Proof by Induction

- Let P be some proposition
- To prove P(n) is true for all n ≥ 0
  - Base case: Prove P(0)
  - Induction case: Let k ≥ 0. Assume P(k) is true. Use this assumption to prove P(k+1).

Selection Sort

```
/**
 * @param array array of integers
 * @param startIndex a valid index into array
 */
public static void selectionSort(int[] array, int startIndex) {
     if(startIndex < array.length) {

          // find smallest element in array[startIndex...n]
          int smallest = indexOfSmallest(array, startIndex);

          // move smallest element to position startIndex
          swap(array, smallest, startIndex);

          // recurse on everything to the right of startIndex
          selectionSort(array, startIndex+1);

     }
}
```

Selection Sort (helper)

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

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

Inductive proof of correctness of Selection Sort (on board)

Complexity of Selection sort using Induction (on board)

Complexity of Selection sort using Induction

- SelectionSort(array, n-1)
  - n = array.length
  - Takes time n(n-1)/2,
  - Results in $O(n^2)$ complexity
- Iterative version of selection sort is in text.

InsertionSort

- Similar: To sort array of n elements:
  - Sort first n-1 elements
  - Insert last element in correct position
- How long to insert new element into sorted list of n elements?

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