Ch 16.1 Sort

Sorting algorithms are used to arrange data into some order.
Our text covers 4 flavors: Bubble sort, Selection sort, Insertion sort, Quicksort
The challenge is to understand how each sort works and how it is different from the others.

Bubble sort
In English: "larger values bubble toward the end of the array with each pass" (page 953)
Wikipedia: en.wikipedia.org/wiki/Bubble_sort
Pseudo-code (page 956):

```
for last position = last subscript in array, decrement down to 0
   for index = 0 up through last position - 1
      if array[index] > array[index + 1]
         swap array[index] with array[index + 1]
      endif
   endfor
endfor
```

Selection sort
In English: "The smallest value in the array is located and moved to position 0. The the next smallest value is located and moved to position 1. This process continues until all of the elements have been placed in their proper order" (page 962)
Wikipedia: en.wikipedia.org/wiki/Selection_sort
Pseudo-code: (www.cs.miami.edu/~burt/learning/Csc517.051/notes/selection.html)

```
SelectionSort(A)
1  n := length[A]
2  for i := 1 to n
3     // GOAL: place the correct number in A[i]
4     j := FindIndexOfSmallest( A, i, n )
5     swap A[i] with A[j]  // A[1..i] the i smallest numbers sorted
6  end-for

FindIndexOfSmallest( A, i, n )
1  smallestAt := i ;
2  for j := (i+1) to n
3     if ( A[j] < A[smallestAt] ) smallestAt := j
      // L.I. A[smallestAt] smallest among A[i..j]
4  end-for
5  return smallestAt
```

Insertion sort

In English: "Insertion sort iterates, consuming one input element each repetition, and growing a sorted output list. Each iteration, insertion sort removes one element from the input data, finds the location it belongs within the sorted list, and inserts it there. It repeats until no input elements remain." (Wikipedia page)

Wikipedia: en.wikipedia.org/wiki/Insertion_sort

Pseudo-code: (wikipedia)

```
for i ← 1 to length(A)
    j ← i
    while j > 0 and A[j-1] > A[j]
        swap A[j] and A[j-1]
        j ← j - 1
```

Quicksort

In English: "Quicksort is a divide and conquer algorithm. Quicksort first divides a large list into two smaller sub-lists: the low elements and the high elements. Quicksort can then recursively sort the sub-lists." (Wikipedia page)

Wikipedia: en.wikipedia.org/wiki/Quicksort

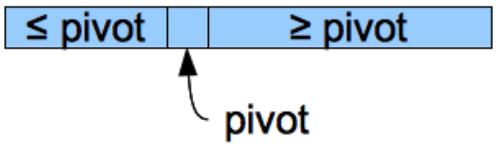
Pseudo-code: (www.algorithmist.com/index.php/Quicksort)

```
Quicksort(A as array, low as int, high as int)
   if (low < high)
      pivot_location = Partition(A,low,high)
   Quicksort(A,low, pivot_location - 1)
   Quicksort(A, pivot_location + 1, high)

Partition(A as array, low as int, high as int)
   pivot = A[low]
   leftwall = low

   for i = low + 1 to high
      if (A[i] < pivot) then
         leftwall = leftwall + 1
         swap(A[i], A[leftwall])

   swap(A[low],A[leftwall])
   return (leftwall)
```



Helpful links:

- www.cs.oswego.edu/~mohammad/classes/csc241/samples/sort/Sort2-E.html - a very nice, straightforward animation of our 4 sorting algorithms
- www.sorting-algorithms.com/ - animation and analysis of our basic sort algorithms
- en.wikipedia.org/wiki/Sorting_algorithm - root Wikipedia page for all sorting algorithms
- www.algorithmist.com/index.php/Sorting - nice, compact summaries