Selection Code

```
private static int indexOfLargest(Comparable[] theArray,
    int size) {
  // ------------------------------------------------
  // Finds the largest item in an array.
  // Precondition: theArray is an array of size items;
  // size >= 1.
  // Postcondition: Returns the index of the largest
  // item in the array.
  // ------------------------------------------------
    int indexSoFar = 0; // index of largest item found so far
  // Invariant: theArray[indexSoFar]>=theArray[0..currIndex-1]
    for (int currIndex = 1; currIndex < size; ++currIndex) {
      if (theArray[currIndex].compareTo(theArray[indexSoFar])>0) {
        indexSoFar = currIndex;
      } // end if
    } // end for
    return indexSoFar; // index of largest item
  } // end indexOfLargest
```

This loop executes a total of

$(n-1) + (n-2) + \cdots + 1 =$

$n * (n-1)/2$ times

Each execution of the indexOfLargest loop perform one comparison(?)

$1 \times n * (n-1)/2$ (b)

From (a) and (b)

$3 * (n-1) + n * (n-1)/2$

$3n - 3 + \frac{n^2}{2} - \frac{n}{2} = \frac{n^2}{2} + \frac{5}{2}n - 3$

$O(n^2)$