Name: \_\_Daniel Grimshaw\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Selection and Insertion Sort Assignment**

**Insertion Sort**

1. An array of type int is to be sorted smallest to largest using the insertionSort method. If the array originally contains

{18, 35, 15, 14, 81, 77, 45, 8}

what will it look like after the first through third passes of the for loop?

Pass 1: \_\_\_\_\_{18, 35, 15, 14, 81, 77, 45, 8}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Pass 2: \_\_\_\_\_{15, 18, 35, 14, 81, 77, 45, 8}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Pass 3: {14, 15, 18, 35, 81, 77, 45, 8}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. An array of Strings is to be sorted alphabetically in descending order (from z to a) using the insertionSort method. If the array originally contains

{Idaho, Mississippi, Alaska, Oregon, Washington, California, Utah, Arizona}

what will it look like after the first through fourth passes of the for loop?

Pass 1: { Mississippi, Idaho, Alaska, Oregon, Washington, California, Utah, Arizona}

Pass 2: { Mississippi, Idaho, Alaska, Oregon, Washington, California, Utah, Arizona}

Pass 3: { Oregon, Mississippi, Idaho, Alaska, Washington, California, Utah, Arizona}

Pass 4: { Washington, Oregon, Mississippi, Idaho, Alaska, California, Utah, Arizona}

3. Modify the insertion sort method below to handle an array of String values rather than an array of int values. Make changes in red to the code below.

**// Sorts a[0] ... a[size-1] in ascending order**

**// using the insertion sort algorithm.**  
public static void insertionSort(String[] a) {  
 for (int n = 1; n < a.length; n++) {  
 **// Save the next element to be inserted.**  
 String aTemp = a[n];  
 **// Going backward from a[n-1], shift**

**// elements to the right until an element**  
 **// a[i] <= aTemp is found.**  
 int i = n;  
 while (i > 0 && aTemp.compareTo(a[i-1]) <= 0 ) {  
 a[i] = a[i-1];  
 i--;  
 }  
 **// Insert the saved element into a[i];**  
 a[i] = aTemp;  
 }  
 }

**Sorts**

4. Consider the task of sorting the elements of an array in ascending order. Which of the following statements are true? (Explain your answer.)

I. Selection Sort always requires more comparisons than Insertion Sort.

II. Insertion Sort always requires more moves than Selection Sort.

III. Insertion Sort, on average, requires more moves than Selection Sort.

(A) I only (B) II only **(C) III only** (D) I and II (E) II and III

Explanation: \_\_\_You have to switch each element.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Given the sort method below and array letters2[] = {"R", "W", "D", "A", "M", "O", "D"}, what does the array look like after the first through fourth passes of the outer for loop?

Pass 1: {"W", "R", "D", "A", "M", "O", "D"}

Pass 2: {"W", "R", "O", "A", "M", "D", "D"}

Pass 3: {"W", "R", "O", "M", "A", "D", "D"}

Pass 4: {"W", "R", "O", "M", "D", "A", "D"}

public static void sort(String[] a) {  
 int pos; String m;  
 for (int i = 0; i < a.length - 1; i++) {  
 m = a[i];  
 pos = i;  
 for (int j = i + 1; j < a.length; j++) {  
 if (m.compareTo(a[j])<0) {  
 m = a[j];  
 pos = j;  
 }  
 }  
 String temp = a[i];  
 a[i] = a[pos];  
 a[pos] = temp;  
 }  
 }

6. Given the sort method below and array String letters[] = {"C", "E", "B", "F", "H", "P", "V", "Z", "A", "C"}, what does the array look like after the first through fourth passes of the outer for loop?

Pass 1: {"C", "E", "B", "F", "H", "P", "V", "C", "A", "Z"}

Pass 2: {"C", "E", "B", "F", "H", "P", "A", "C", "V", "Z"}

Pass 3: {"C", "E", "B", "F", "H", "C", "A", "P", "V", "Z"}

Pass 4: {"C", "E", "B", "F", "A", "C", "H", "P", "V", "Z"}

public static void sort(String[] a) {  
 int pos;  
 for (int i = a.length; i >= 2; i--) {  
 // find index of smallest remaining value  
 pos = 0;  
 for (int j = 1; j < i; j++) {  
 if (a[j].compareTo(a[pos])>0) {  
 pos = j;  
 }  
 }  
 // swap smallest value its proper place, a[i]  
 swap(a, i-1, pos);  
 }  
 }  
   
 public static void swap(String[] a, int i, int pos){  
 String temp = a[i];  
 a[i] = a[pos];  
 a[pos] = temp;  
 }