MULTIPLE-CHOICE QUESTIONS ON SORTING AND SEARCHING

1. The decision to choose a particular sorting algorithm should be made based on
   I. Run-time efficiency of the sort
   II. Size of the array
   III. Space efficiency of the algorithm
   (A) I only
   (B) II only
   (C) III only
   (D) I and II only
   (E) I, II, and III

2. The following code fragment does a sequential search to determine whether a
   given integer, value, is stored in an array a[0] ... a[n-1].

   ```c
   int i = 0;
   while (/* boolean expression */)
   {
     i++;
   }
   if (i == n)
     return -1;  // value not found
   else
     return i;   // value found at location i
   ```

   Which of the following should replace /* boolean expression */ so that the algo-
   rithm works as intended?
   (A) value != a[i]
   (B) i < n && value == a[i]
   (C) value != a[i] && i < n
   (D) i < n && value != a[i]
   (E) i < n || value != a[i]

3. A feature of data that is used for a binary search but not necessarily used for a
   sequential search is
   (A) length of list.
   (B) type of data.
   (C) order of data.
   (D) smallest value in the list.
   (E) median value of the data.
4. Array `unsortedArr` contains an unsorted list of integers. Array `sortedArr` contains a sorted list of integers. Which of the following operations is more efficient for `sortedArr` than `unsortedArr`? Assume the most efficient algorithms are used.

I. Inserting a new element  
II. Searching for a given element  
III. Computing the mean of the elements

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

5. An algorithm for searching a large sorted array for a specific value `x` compares every third item in the array to `x` until it finds one that is greater than or equal to `x`. When a larger value is found, the algorithm compares `x` to the previous two items. If the array is sorted in increasing order, which of the following describes all cases when this algorithm uses fewer comparisons to find `x` than would a binary search?

(A) It will never use fewer comparisons. 
(B) When `x` is in the middle position of the array 
(C) When `x` is very close to the beginning of the array 
(D) When `x` is very close to the end of the array 
(E) When `x` is not in the array

6. Assume that `a[0] ... a[N-1]` is an array of `N` positive integers and that the following assertion is true:

\[ a[0] > a[k] \text{ for all } k \text{ such that } 0 < k < N \]

Which of the following must be true?

(A) The array is sorted in ascending order.  
(B) The array is sorted in descending order.  
(C) All values in the array are different.  
(D) `a[0]` holds the smallest value in the array.  
(E) `a[0]` holds the largest value in the array.

7. The following code is designed to set `index` to the location of the first occurrence of `key` in array `a` and to set `index` to `-1` if `key` is not in `a`.

```java
index = 0;
while (a[index] != key)
    index++;
if (a[index] != key)
    index = -1;
```

In which case will this program definitely fail to perform the task described?

(A) When `key` is the first element of the array  
(B) When `key` is the last element of the array  
(C) When `key` is not in the array  
(D) When `key` equals 0  
(E) When `key` equals `a[key]`
8. Refer to method search.

```java
/* Precondition: v[0]...v[v.length-1] are initialized.
 * Postcondition: Returns k such that -1 <= k <= v.length-1.
 * If k >= 0 then v[k] == key. If k == -1,
 * then key != any of the elements in v. */
public static int search(int[] v, int key)
{
    int index = 0;
    while (index < v.length && v[index] < key)
        index++;
    if (index != v.length)
        return index;
    else
        return -1;
}
```

Assuming that the method works as intended, which of the following should be added to the precondition of search?

(A) v is sorted smallest to largest.
(B) v is sorted largest to smallest.
(C) v is unsorted.
(D) There is at least one occurrence of key in v.
(E) key occurs no more than once in v.

Questions 9-14 are based on the binSearch method and the private instance variable a for some class:

```java
private int[] a;

/* Does binary search for key in array a[0]...a[a.length-1],
 * sorted in ascending order.
 * Postcondition: Returns index such that a[index]=key.
 * If key not in a, returns -1. */
public int binSearch(int key)
{
    int low = 0;
    int high = a.length - 1;
    while (low <= high)
    {
        int mid = (low + high) / 2;
        if (a[mid] == key)
            return mid;
        else if (a[mid] < key)
            low = mid + 1;
        else
            high = mid - 1;
    }
    return -1;
}
```

A binary search will be performed on the following list:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>20</td>
<td>24</td>
<td>30</td>
<td>41</td>
</tr>
</tbody>
</table>
9. To find the key value 27, the search interval after the first pass through the while loop will be
   (A) a[0] ... a[7]
   (B) a[5] ... a[6]
   (C) a[4] ... a[7]
   (D) a[2] ... a[7]
   (E) a[6] ... a[7]

10. How many iterations will be required to determine that 27 is not in the list?
    (A) 1
    (B) 3
    (C) 8
    (D) 27
    (E) An infinite loop since 27 is not found

11. What will be stored in y after executing the following?
    ```
    int y = binSearch(4);
    ```
    (A) 20
    (B) 7
    (C) 4
    (D) 0
    (E) -1

12. If the test for the while loop is changed to
    ```
    while (low < high)
    ```
    the binSearch method does not work as intended. Which value in the given list will not be found?
    (A) 4
    (B) 7
    (C) 11
    (D) 24
    (E) 30

13. For binSearch, which of the following assertions will be true following every iteration of the while loop?
    (A) key = a[mid] or key is not in a.
    (B) a[low] ≤ key ≤ a[high]
    (C) low ≤ mid ≤ high
    (D) key = a[mid], or a[low] ≤ key ≤ a[high]
    (E) key = a[mid], or a[low] ≤ key ≤ a[high], or key is not in array a.

14. Suppose n = a.length. A loop invariant for the while loop is: key is not in array a, or
    ```
    (A) a[low] < key < a[high], 0 ≤ low ≤ high+1 ≤ n
    (B) a[low] < key < a[high], 0 ≤ low ≤ high+1 ≤ n
    (C) a[low] < key < a[high], 0 ≤ low ≤ high ≤ n
    (D) a[low] < key < a[high], 0 ≤ low ≤ high ≤ n
    (E) a[low] < key < a[high], 0 ≤ low ≤ high ≤ n-1
    ```
For Questions 15-19 refer to the `insertionSort` method and the private instance variable `a`, both in a Sorter class.

```java
private int[] a;

/* Precondition: a[0], a[1]...a[a.length-1] is an unsorted array
   * of `Comparable` objects.
   * Postcondition: Array `a` is sorted in descending order. */
public void insertionSort()
{
    for (int i = 1; i < a.length; i++)
    {
        int temp = a[i];
        int j = i - 1;
        while (j >= 0 && temp > a[j])
        {
            a[j+1] = a[j];
            j--;
        }
        a[j+1] = temp;
    }
}
```

15. An array of `Integer` is to be sorted biggest to smallest using the `insertionSort` method. If the array originally contains

```
    1 7 9 5 4 12
```

what will it look like after the third pass of the `for` loop?

(A) 9 7 1 5 4 12  
(B) 9 7 5 1 4 12  
(C) 12 9 7 1 5 4  
(D) 12 9 7 5 4 1 
(E) 9 7 12 5 4 1 

16. When sorted biggest to smallest with `insertionSort`, which list will need the fewest changes of position for individual elements?

(A) 5, 1, 2, 3, 4, 9 
(B) 9, 5, 1, 4, 3, 2 
(C) 9, 4, 2, 5, 1, 3 
(D) 9, 3, 5, 1, 4, 2 
(E) 3, 2, 1, 9, 5, 4 

17. When sorted biggest to smallest with `insertionSort`, which list will need the greatest number of changes in position?

(A) 5, 1, 2, 3, 4, 7, 6, 9 
(B) 9, 5, 1, 4, 3, 2, 1, 0 
(C) 9, 4, 6, 2, 1, 5, 1, 3 
(D) 9, 6, 9, 5, 6, 7, 2, 0 
(E) 3, 2, 1, 0, 9, 6, 5, 4
22. The elements in a long list of integers are roughly sorted in decreasing order. No more than 5 percent of the elements are out of order. Which of the following is a valid reason for using an insertion sort rather than a selection sort to sort this list into decreasing order?

I. There will be fewer comparisons of elements for insertion sort.
II. There will be fewer changes of position of elements for insertion sort.
III. There will be less space required for insertion sort.

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

23. The code shown sorts array \(a[0] \ldots a[a.length-1]\) in descending order.

```java
public static void sort(int[] a)
{
    for (int i = 0; i < a.length - 1; i++)
        for (int j = 0; j < a.length - i - 1; j++)
            if (a[j] < a[j+1])
                swap(a, j, j + 1);  //swap a[j] and a[j+1]
}
```

This is an example of

(A) selection sort.
(B) insertion sort.
(C) mergesort.
(D) quicksort.
(E) none of the above.

24. Which of the following is a valid reason why mergesort is a better sorting algorithm than insertion sort for sorting long lists?

I. Mergesort requires less code than insertion sort.
II. Mergesort requires less storage space than insertion sort.
III. Mergesort runs faster than insertion sort.

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

25. A large array of lowercase characters is to be searched for the pattern “pqr.” The first step in a very efficient searching algorithm is to look at characters with index

(A) 0, 1, 2, ... until a “p” is encountered
(B) 0, 1, 2, ... until any letter in “p” ... “s” is encountered
(C) 3, 7, 11, ... until an “s” is encountered
(D) 3, 7, 11, ... until any letter in “p” ... “s” is encountered
(E) 3, 7, 11, ... until any letter other than “p” ... “s” is encountered
30. A binary search is to be performed on an array with 600 elements. In the worst case, which of the following best approximates the number of iterations of the algorithm?
   (A) 6
   (B) 10
   (C) 100
   (D) 300
   (E) 600

31. A worst case situation for insertion sort would be
   I A list in correct sorted order.
   II A list sorted in reverse order.
   III A list in random order.
   (A) I only
   (B) II only
   (C) III only
   (D) I and II only
   (E) II and III only

32. Which of the following represents a heap?

(AB ONLY)

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

33. The list
   17 9 2 7 21 18 4 5
   is to be sorted into ascending order using heapsort. What is the level of the binary tree that will be formed, given that the root is at level 0?
   (A) 0
   (B) 1
   (C) 2
   (D) 3
   (E) 4
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>E</td>
</tr>
<tr>
<td>2</td>
<td>D</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
</tr>
<tr>
<td>5</td>
<td>C</td>
</tr>
<tr>
<td>6</td>
<td>E</td>
</tr>
<tr>
<td>7</td>
<td>C</td>
</tr>
<tr>
<td>8</td>
<td>A</td>
</tr>
<tr>
<td>9</td>
<td>C</td>
</tr>
<tr>
<td>10</td>
<td>B</td>
</tr>
<tr>
<td>11</td>
<td>D</td>
</tr>
<tr>
<td>12</td>
<td>A</td>
</tr>
<tr>
<td>13</td>
<td>E</td>
</tr>
<tr>
<td>15</td>
<td>B</td>
</tr>
<tr>
<td>16</td>
<td>B</td>
</tr>
<tr>
<td>17</td>
<td>A</td>
</tr>
<tr>
<td>22</td>
<td>A</td>
</tr>
<tr>
<td>23</td>
<td>E-bubble sort!</td>
</tr>
<tr>
<td>25</td>
<td>D</td>
</tr>
<tr>
<td>30</td>
<td>B</td>
</tr>
<tr>
<td>31</td>
<td>B</td>
</tr>
</tbody>
</table>

21