**Assignment 2**

**RDC 10/10/2017**

**Name:\_\_\_\_\_\_\_\_\_\_\_\_ Class:\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1. Complete the following code segment.**

/\*\* @return index of smaller element in array arr of integers \*/

public static int findMin (int[] arr)

{

int min = arr[0];

int minIndex = 0;

for (int i = 1; i < arr.length; i++)

if (arr[i] < min) //found a smaller element

{

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

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

}

return minIndex;

}

**2. Complete the following code segment.**

/\*\* Add 3 to each element of array b. \*/

public static void changeArray (int[] b)

{

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ //Tips: use loop structure

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

}

( ) **3. Which of the following correctly initializes an array arr to contain four elements each with value 0?**

[a] int[] arr = {0, 0, 0, 0};

[b] int[] arr = new int[4];

[c] int[] arr = new int[4];

for (int i = 0; i < arr.length; i++)

arr[i] = 0;

(A) [a] only

(B) [c] only

(C) [a] and [c] only

(D) [b] and [c] only

(E) [a], [b] and [c]

**4. Refer to the following code segment. You may assume that array arr1 contains elements arr1[0], arr1[1], …, arr1[N-1], where N = arr1.length.**

int count = 0;

for (int i = 0; i < N; i++)

if (arr1[i] != 0)

{

arr1[count] = arr1[i];

count++;

}

int[] arr2 = new int[count];

for (int i = 0; i < count; i++)

arr2[i] = arr1[i];

**If array arr1 initially contains the elements 0, 6, 0, 4, 0, 0, 2 in this order, what will arr2 contain after execution of the code segment? Try to solve this problem without running the codes.**

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

( ) **5. The following code fragment does a sequential search to determine whether a given integer, value, is sorted in an array a[0] … a[n-1].**

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 algorithm work 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]

( ) **6. A feature of data that is used for a binary search but not necessarily 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

**For Questions 7-9 refer to the insertionSort method and the private instance variable a, both in a Sorter class.**

private Integer[] a;

/\*\* **Precondition**: a[0], a[1]…a[a.length-1] is an unsorted array of

\* Integer objects.

\* **Postcondition**: Array a is sorted in descending order.

\*/

public void insertionSort()

{

for (int i = 1; i < a.length; i++)

{

Integer temp = a[i];

int j = i - 1;

while (j >= 0 && temp.compareTo(a[j]) > 0)

{

a[j+1] = a[j];

j = j - 1;

}

a[j+1] = temp;

}

}

( ) **7. 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

( ) **8. 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

( ) **9. When sorted biggest to smallest with insertionSort, which list will need the greatest changes of position?**

(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

**Bonus**

**1. Read the following blogs.**

<http://blog.csdn.net/u011464124/article/details/70054204>

<http://blog.csdn.net/xiazdong/article/details/8462393>