# Unit 3 Discussion: Searching and Sorting; 2D Array

1. What was the hardest part of Unit 2 assignments? Please explain. (Note: the purpose of this discussion question is for you to reflect on your assignments last week and share some tips <what failed, what worked, …> with the class)

2. What did you find most confusing or difficult about what you read this week? You may also ask for help on a PA/CA or use of zyBook. (Note: be specific. For example, instead of “xx is hard”, identify the topic and illustrate with an example.)

3. Rewrite the linearSearch() method in Figure 17.1.1 (ch17.1) to have it search for a string in an array of strings. Use case sensitive comparison. Modify the main() as well to test the method.

4. For the binarySearch method in Figure 17.2.1 (ch17.2), what is low and high after the first iteration of the while loop (i.e. after the body of the while loop is executed once) when invoking:

int[] x = new int[]{1, 4, 6, 8, 10, 15, 20};

binarySearch(x, 11);

Feel free to use tools like Java Visualizer.

A. low is 0 and high is 6

B. low is 0 and high is 3

C. low is 3 and high is 6

D. low is 4 and high is 6

E. low is 0 and high is 5

5. Rewrite the binarySearch() method in Figure 17.2.1 (ch17.2) to have it search for a string in an array of strings. Use case sensitive comparison. Modify the main() as well to test the method.

6. Answer those questions about binary search:

a) explain in your own words that why binary search is preferred over linear search when the array is ordered.

b.) Can we use binary search if we need to find the first occurrence of a target value in an array with duplicate values? Illustrate using this array {4, 5, 5, 8, 8, 8, 9, 11, 11, 21, 22, 56, 56, 78}

7. What’s the content of the array object after the loop ends?

public class Test {

public static void main(String[] args) {

int list[] = {1, 2, 3, 4, 5, 6};

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

list[i] = list[i - 1];

// What’s the content of array list pointed by list?

}

}

A. 1 2 3 4 5 6

B. 2 3 4 5 6 6

C. 2 3 4 5 6 1

D. 1 1 1 1 1 1

8. How many times longer does sorting a list of 40 elements take compared to a list of 5 elements? Selection sort will be used.

A. 32

B. 5

C. 8

D. 64

9. The insertion sort code explained in the lecture PPT is slightly different than the one in the textbook (Figure 17.7.1 of ch17.7). Explain how those two versions of the method may behave differently.

10. Write a public static method named sortDec that receives an array of int arr and sorts the array into descending order (i.e. arr[0] >= arr[1] >= … >= arr[arr.length-1]). Write your own sorting code and do not call a Java API sorting method. Do not first sort the array into ascending order and then reverse.

11. Write a method to merge two sorted arrays into one.

// Before the method starts, arr1 and arr2 are both sorted and in ascending order

// This method should return an array that holds all elements from arr1 and arr2 and that are in ascending order

public static int[] merge(int[] list1, int[] list2) {

// ADD CODE

/\* You may copy two arrays into one big array one after the other

and then sort the big array, but that’s not very efficient as

it doesn’t take advantage of the “already sorted” property.

A better approach is to scan the two arrays at the same time and each time compare 2 elements, one from each arr, and copy the smaller one into the big array

For example, given {>1, 5, 7, 10, 12, 14} and {>2, 4, 10}, we start from first pair 1 and 2. 1 is smaller so is copied into result.

Next {1, >5, 7, 10, 12, 14} and {>2, 4, 10} i.e. pair 5 and 2. 2 is smaller and copied.

Next {1, >5, 7, 10, 12, 14} and {2, >4, 10} i.e. pair 5 and 4, and 4 is copied.

Next {1, >5, 7, 10, 12, 14} and {2, 4, >10} i.e. pair 5 and 10, and 5 is copied.

Next {1, 5, >7, 10, 12, 14} and {2, 4, >10} i.e. pair 7 and 10, and 7 is copied.

Next {1, 5, 7, >10, 12, 14} and {2, 4, >10} i.e. pair 10 and 10. Equal, so one is copied (assume 10 from list1).

Next {>1, 5, 7, 10, >12, 14} and {2, 4, >10} i.e. pair 12 and 10, and 10 is copied.

Now {1, 5, 7, 10, >12, 14} and {2, 4, 10>}. We’ve finished scanning list2. The rest of list1 should be directly appended to the big list.

\*/

} // end merge

12. What is the content of 2D array x after the code segment has been executed?

int[][] x = new int[3][4];

for (int row = 0; row < x.length; row++)

for (int col = 0; col < x[0].length; col++)

if (row < col)

x[row][col] = 1;

else if (row == col)

x[row][col] = 2;

else

x[row][col] = 3;

A.

{{2, 1, 1},

{3, 2, 1},

{3, 3, 2},

{3, 3, 3})

B.

{{2, 3, 3},

{1, 2, 3},

{1, 1, 2},

{1, 1, 1})

C.

{{2, 3, 3, 3},

{1, 2, 3, 3},

{1, 1, 2, 3}}

D.

{{2, 1, 1, 1},

{3, 2, 1, 1},

{3, 3, 2, 1}}

E.

{{1, 1, 1, 1},

{2, 2, 2, 2},

{3, 3, 3, 3}}

13. Given a 2D array element arr2d[row][col] (row and col can be any valid index), specify the index of each of its 8 neighboring elements.

For example, arr2d[2][2] and its neighbors | arr2d[1][4] and its neighbors

1 2 3 4 5 6 1 2 3 4 5 6

11 12 13 14 15 16 11 12 13 14 **15** 16

21 22 **23** 24 25 26 21 22 23 24 25 26

31 32 33 34 35 36 31 32 33 34 35 36

14. Write a method to find the row index of the maximum value in a specified column of a given a 2D array. Return -1 if an invalid column value is specified. You may test your code with the Test2DArr.java given in this unit.

public static int findRowOfColMax(int[][] arr2d, int col){

// ADD code

} // end findRowOfColMax

It may be used like this:

**int** row = *findRowOfColMax*(arr2d, col);

System.***out***.println("The largest element in column " + col + " is " + arr2d[row][col] + " from row " + row);

15. Write a method to find the col index of the minimum value in a specified row of a given a 2D array. Return -1 if an invalid row value is specified. You may test your code with the Test2DArr.java given in this unit.

public static int findColOfRowMin(int[][] arr2d, int row) {

// ADD code

} // end findColOfRowMin

16. Write a method to switch two rows of a given 2D array. For example, a call of switchTwoRows(arr2d, row1, row2) would swap row1 of the 2D array with row2.

public static void switchTwoRows(int[][] x, int row1, int row2) {

// ADD code

} // end switchTwoRows

17. Write a method named getMask that receives a single parameter named table, which is a two-dimensional array of int. The getMask method should create and return an array *mask* for the passed-in table array. (The programming term *mask* refers to an array that is built from another array and it contains all 0’s and 1’s.) Assume the parameter has at least one row and one column. For each element in the mask array, if the original array’s corresponding element contains a positive number, the mask array’s element should contain a 1. And if the original array’s corresponding element contains a zero or negative number, the mask array’s element should contain a 0. Note this example:

table parameter returned array

5 -2 3 1 1 0 1 1

0 14 0 6 0 1 0 1

3 6 -1 4 1 1 0 1

Note:

* Your method should not change the content of the parameter array.
* Your method should work with a 2d array parameter of any dimensions, not just 3-row, 4-column tables.