Student Performance Q&A:

2015 AP® Computer Science A Free-Response Questions

The following comments on the 2015 free-response questions for AP® Computer Science A were written by the Chief Reader, Dr. Elizabeth Johnson, Xavier University. They give an overview of each free-response question and of how students performed on the question, including typical student errors. General comments regarding the skills and content that students frequently have the most problems with are included. Some suggestions for improving student performance in these areas are also provided. Teachers are encouraged to attend a College Board workshop to learn strategies for improving student performance in specific areas.

Question 1

What was the intent of this question?

This question focused on accessing and processing data from one and two-dimensional arrays. It involved abstraction through the use of methods. Part (a) asked students to sum the integers stored in a one-dimensional array. Part (b) asked students to sum the rows of a two-dimensional array by explicitly calling the method from part (a). Part (c) asked students to call the method from part (b) to obtain the sums of the rows of a two-dimensional array and analyze those sums. More specifically, it asked students to look for duplicates within the one-dimensional array of sums and to return the answer `false` if the one-dimensional array had any duplicate sums and `true` if it did not have any duplicate sums.

How well did students perform on this question?

Of the four free-response questions this year, students performed best on this question. The mean score was 5.05 out of a possible 9 points with a standard deviation of 3.36.

What were common student errors or omissions?

Students had difficulty with declaring and constructing arrays of the correct size. They also failed to initialize variables before use and forgot to return the values they computed. In general, students also struggled with the relationship of one-dimensional and two-dimensional arrays.

Students rewrote code rather than using a previously defined function (as instructed in the problem). In part (c) students made conceptual errors in implementing the algorithm to check for the presence of equal sums. Many students did not realize that two loops were needed to do all comparisons. For example, they used one loop and compared only adjacent elements.

Based on your experience of student responses at the AP® Reading, what message would you like to send to teachers that might help them to improve the performance of their students on the exam?

Teachers should review enhanced `for` loops. Teachers should review two-dimensional arrays in general as well as how to access a one-dimensional array that represents a row in the two-dimensional array. Students need to understand how to use previously defined methods to simplify problems. Students also need to learn and practice common collection algorithms such as how to sum values and how to find a value in a collection.

Question 2

What was the intent of this question?

This question focused on class design and the `String` class. Students had to demonstrate an understanding of class, constructor, and method header syntax. Students were required to understand the importance of declaring instance variables `private` and initializing instance variables in a constructor. Students were also required to demonstrate an understanding of `String` methods.

Students were asked to write a class that represents a hidden word and includes a method to return a hint for a given guess. To create the hint string, students had to traverse the hidden word and the guess by comparing each letter, one by one, and adding a character to the resulting hint string. The character was the letter if the guess letter and the hidden word letter matched and were in the same position. The character was a `"+"` if the guess letter and the hidden word letter matched and were in different positions. The character was a `"*"` if the guess letter and the hidden word letter did not match. The resulting hint string was then returned.

How well did students perform on this question?

This question appears to have been slightly more difficult than question 1 but not as difficult as question 3 or 4. The mean score was 4.41 out of a possible 9 points with a standard deviation of 3.40.

What were common student errors or omissions?

Declaration of parameters was a common source of error. Students declared method parameters as local variables, instance variables as constructor parameters, and method parameters as instance variables. Students had difficulty using String methods such as `indexOf and substring`. Students often assigned a value to a method call. For example, statements such as `object.substring(i,i+1) = "*"` appeared in code.

In testing for equality of objects and primitives, students were often confused about whether to use `==` or the `equals` method.

Based on your experience of student responses at the AP® Reading, what message would you like to send to teachers that might help them to improve the performance of their students on the exam?

More practice is needed working with, traversing, comparing, and constructing `Strings.` Students need to be more familiar with the syntax for `String` methods in the subset. Students need to practice using the quick reference guide in the exam booklet to clarify library method use. Students also need practice in designing classes to specification.

Question 3

What was the intent of this question?

This question involved the use of the `ArrayList` data structure, `ArrayList` traversal, and both the access and modification of `ArrayList` elements. The question also involved modification of a private instance variable. Students were provided with the specifications of two classes: `SparseArrayEntry` and `SparseArray`. The `SparseArray` class represents a sparse array. It contains a list, `entries`, of `SparseArrayEntry` objects that represent non-zero elements in a sparse array. Students were required to call methods from `SparseArrayEntry`.

In part (a) students were asked to write the `SparseArray` method `getValueAt(int row, int col)`, which returns the value of the sparse array element at the given row and column. If no such element exists in the `entries ArrayList`, then the method returns `0`. In part (b) students were asked to write the `SparseArray` method `removeColumn(int col)` to accomplish three tasks. First, the method removes all elements in `entries` with column index equal to the `col` parameter. Second, the method replaces all elements having a column index greater than the `col` parameter with elements having column indexes decremented by one. Finally, the method decrements the `numCols` instance variable by one to reflect the new dimension of the sparse array.

How well did students perform on this question?

This question was the most difficult on the exam for students. The mean score was 3.05 out of a possible 9 points with a standard deviation of 3.40. Many students received 0 points for this question.

What were common student errors or omissions?

The major student error on this question was mistaking the problem for a two-dimensional array problem. Many students did not understand the abstraction wherein only non-zero elements of the sparse array were included in an `ArrayList`. Students attempted instead to access elements as if they were stored in a two-dimensional array. Other errors included early returns in a loop during a search, reset of a return value within the loop, and attempting to remove elements in an enhanced `for` loop over the `ArrayList`. Students also made errors in logic when using `if` statements by using consecutive `if` statements rather than nested `if-else if` statements. Students also had difficulty creating new sparse array entries and adjusting the column numbers correctly.

Based on your experience of student responses at the AP® Reading, what message would you like to send to teachers that might help them to improve the performance of their students on the exam?

Teachers should have students work on reading strategies to interpret questions as well as practice code reading without compilers. The abstraction in this question was difficult for students, and they need more practice in visualizing the data structure being described in the question. Students need to understand that row and column may be used in contexts other than two-dimensional arrays. Students need to review removal of multiple elements in a list. Teachers should also discuss the impact of sequential `if` statements versus nested `if-else if` statements and when each is appropriate.

Question 4

What was the intent of this question?

This question asked students to write an interface specification, a complete class implementing that interface, and the body of a method in another class implementing the interface. In part (a) students were asked to write the `NumberGroup` interface. A `NumberGroup` represents a collection of integers. Its single required method is `contains`, which takes as its parameter an integer and returns `true` when that integer is part of the collection and `false` otherwise. The students needed to know the correct syntax for the interface declaration and how to provide exactly one method declaration inside that interface. It is particularly important that the students not attempt to provide any method implementation.

In part (b) students were asked to write a `Range` class that implements the `NumberGroup` interface. Students were expected to provide a constructor with two integer parameters and private instance variables to maintain enough object state for the `contains` method to work. Two kinds of approaches work equally well here: saving the minimum and maximum values of the range; or building an array or `ArrayList` containing all the elements of the range. Students also needed to write the `contains` method that was specified by the `NumberGroup` interface.

In part (c) students were asked to implement the `contains` method of the `MultipleGroups` class. The `MultipleGroups` class has a single `List` instance variable `groupList`, which is used to store a collection of `NumberGroup` objects. The method takes an integer and returns `true` if and only if the integer is contained in one or more of the number groups in `groupList`.

How well did students perform on this question?

This question was the second most difficult on the exam for students. The mean score was 3.99 out of a possible 9 points with a standard deviation of 3.19.

What were common student errors or omissions?

This was the first time the students have been asked to write an interface on the exam, and many students attempted to write the body of the `contains` method in the interface. There was confusion as to where to define the instance variables. Most students confused the interface with an abstract class.

In part (b) a number of students used the keyword `extends` instead of `implements`. Students also neglected to define the instance variables as private. Other student errors included assignment dyslexia, failure to instantiate the array/`ArrayList` and loop bounds errors. The most common error was the failure to correctly implement the `contains` method. Many omitted it entirely.

In part (c) students were asked to write the `contains` method of the `MultipleGroups` class. Many students assumed that all of the elements in `groupList` were of the `Range` class. While writing that class in part (b), students created access methods and then attempted to use them in part (c). In doing so, they neglected to call the `contains` method on the elements of `MultipleGroups`.

Based on your experience of student responses at the AP® Reading, what message would you like to send to teachers that might help them to improve the performance of their students on the exam?

Teachers are advised to review the requirements of an interface and stress the differences between an interface and an abstract class. In particular, teachers should emphasize that interfaces should only contain method declarations that are to be defined in classes that implement them. Students should also practice writing multiple classes that implement the same interface.