Student Performance Q&A:

2016 AP® Computer Science A Free-Response Questions

The following comments on the 2016 free-response questions for AP® Computer Science A were written by the Chief Reader, Elizabeth Johnson of 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 class design, inheritance, the use of the array data structure, and the accessing of elements from that array.

In part (a) students were asked to write the complete `RandomStringChooser` class. Students had to demonstrate an understanding of class, constructor, and method header syntax. Students were expected to choose appropriate instance variable(s) to maintain object state, declaring those instance variables `private`, and initializing the instance variables in a constructor. Students needed to include in their class a `getNext` method, which returns a randomly chosen `String` from the array argument, taking care to not return a particular `String` more than once. Two approaches work equally well here: making a second array to help keep track of used elements or copying all of the array elements into an `ArrayList` and removing elements from that list after they are used. Students were explicitly told not to alter the parameter passed to the constructor; therefore, the choice of data structure was significant. Students were also required to demonstrate an understanding of how to generate a random number in the proper range for the purposes of selecting a `String` from the array.

In part (b) students were asked to the write the constructor for the `RandomLetterChooser` class, which is a subclass of the `RandomStringChooser` class they wrote in part (a). Students needed to know how to invoke the constructor from the superclass using `super`. Students were given access to the static `getSingleLetters` method and were explicitly told to use it for full credit rather than reimplement the functionality the method provides. Students were expected to call the `getSingleLetters` method appropriately to create an array of `Strings` from the given `String` parameter.

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 3.42 out of a possible 9 points with a standard deviation of 2.97.

What were common student errors or omissions?

In part (a), the declaration and instantiation of the instance variables was a common source of error. Some students didn't make their instance variable(s) private. Students who chose to use an `ArrayList` often forgot to instantiate an `ArrayList` before adding elements to the list.

The most common student error on this question was made by students who realized they had to update the state of the instance variable, but altered the parameter array passed to the constructor. Students could have avoided this by making a copy of the array, making a new array to hold values that would indicate if the `Strings` were available to them or not, or copying the elements into an `ArrayList`. Instead, they assigned the parameter array to an instance array variable, thus creating a new reference to the parameter array and any changes made to the instance variable were actually being done to the parameter. There was a fair amount of array/`ArrayList` access confusion and many students tried to call a nonexistent `remove` method on an array.

Some students didn't understand that the `getNext` method would have access to the instance variable so they passed the instance variable to the `getNext` method. Finally, when generating random numbers, students were often off by one in the range of numbers produced. Some students generated a `double` and not the `int` that was required.

In part (b), students didn't know that the call to `super` had to be in the first line in the constructor. Other student errors included invoking `getSingleLetters` on the `this` keyword in the call to `super` or on the `String` parameter itself.

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?

Students are very accustomed to declaring instance variables to be of the same type as the given parameters. Have students practice choosing instance variables that will help them solve the problem at hand rather than just matching the parameter type. The `ArrayList` solution was significantly more straightforward than the one using a parallel array, but it was not commonly used. Also have students practice using both arrays and `ArrayLists` in the same problem as many mixed up the syntax when trying to access and/or modify elements. Students also need to understand the implications of making a copy of an array reference as opposed to making a copy of an array.

Question 2

What was the intent of this question?

This question tested the students' ability to use `String` methods from the AP Java subset to perform processing of both `String` parameters and instance variables. This problem also involved interacting classes.

In part (a) students were asked to write a constructor that split a `String` into two parts based on the colon position and initialized instance variables with those parts.

In part (b) students were asked to write a method to test a `String` for certain characteristics and return `true` or `false` depending on the result of those tests.

In part (c) students were asked to write a method to create, fill, and return a new list. They needed to traverse an existing list, use a previously implemented method to identify certain elements, and then delete the identified elements from the original list while adding them to a new list.

How well did students perform on this question?

The mean score was 2.94 out of a possible 9 points with a standard deviation of 2.73.

What were common student errors or omissions?

In part (a), many students utilized example data instead of using parameters. In the constructor implementation they often initialized locally declared variables instead of the instance variables. Their constructor implementations sometimes contained extraneous print statements and `return` statements.

In part (b), difficulties during string manipulation included using `==` instead of `equals` to compare strings for equality, using `substring` parameters that were out of bounds, and treating strings obtained by `substring` as `boolean` values in `if` statement conditions. Also, few solutions included code for all of the different ways in which a string could contain a space-delimited keyword. Most solutions only examined one occurrence of `keyword`, missing situations in which a subsequent `keyword` occurrence was the only one that was properly-delimited.

In part (c), the majority of students either did not attempt to create the required new `ArrayList`, or they attempted to instantiate a `List` interface object with `new List<LogMessage>()`. When traversing a list, they frequently did not access all the elements due to bounds errors, skipped elements after a removal, or attempted a removal in an enhanced `for` (for-each) loop, which causes an exception to be thrown.

Traversing a list backwards is a good technique to avoid skipping elements after removals. However, traversing `messageList` backwards in this problem requires inserting elements at the front of the new list to account for the backwards traversal. Students often added the removed elements to the back of their new list instead, resulting in an incorrectly ordered list.

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?

Have students practice writing constructors and methods which utilize parameters and instance variables.

Provide frequent opportunities for students to use all of the `String` methods from the AP Java subset, including both the one- and two-parameter `substring` methods. In practicing use of `substring`, students should ensure that they capture the desired portion of the string and that they are appropriately protected from bounds errors. Students should learn how to use the `indexOf` method to locate a substring inside of a string, instead of checking character by character. They should be encouraged to use the `substring` method to create one-character strings instead of using `charAt`. This will prevent many `char`-related errors such as attempting to call methods with characters or comparing characters with strings.

Give students ample practice creating and manipulating lists including techniques to remove selected objects without skipping subsequent elements. Students need substantial practice traversing lists and strings both forward and backwards with particular attention to loop bounds.

Question 3

What was the intent of this question?

This question used a two-dimensional (2-D) array of objects to represent a crossword puzzle.

In part (a) students were asked to write a `boolean` method that determines whether or not a specific square in the puzzle should be numbered based on specified labeling rules. Students needed to demonstrate an understanding of `boolean` data structures, writing and evaluating `boolean` expressions, and returning correct `boolean` values. Students were also required to demonstrate an understanding of bounds checking and indexing in a 2-D array. This logic needed to be implemented utilizing given method parameters.

In part (b) students were asked to write a constructor that initializes the instance variable and explicitly calls the method defined in part (a) to build a puzzle, square by square. Students were required to demonstrate an understanding of object instantiation by initializing the class instance variable and a `Square` object using appropriate parameter values. Students were required to demonstrate an understanding of 2-D array processing by traversing a 2-D array in row-major order, accessing each position of the array without going out of bounds. Students were also required to identify squares that needed to be consecutively numbered by calling the previously defined `toBeLabeled` method using appropriate parameters.

How well did students perform on this question?

Of the four free response-questions this year, students performed worst on this question. The mean score was 2.59 out of a possible 9 points with a standard deviation of 2.97.

What were common student errors or omissions?

In part (a), some students failed to check whether the square indexed by the given row and column parameters was white (and thus potentially to be labeled). Many students checked the square above or to the left of the targeted square without checking whether the targeted square was in row 0 or column 0, resulting in a bounds error. A surprising number of students implemented a nested loop, examining many squares rather than just the targeted one. Additionally, students misused logical operators, resulting in incorrect return values.

In part (b), common errors included using incorrect column bounds on both the puzzle instantiation and the inner loop condition. Calling `toBeLabeled` on another object or with incorrect parameters was a frequent error as was omitting the keyword `new` when instantiating an object. Also, some students failed to initialize the instance variable, others re-declared the instance variable, and others declared and populated a local 2-D array.

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?

Have students practice using boolean expressions with logical operators. They also need more practice in instantiating, traversing, and accessing 2-D arrays, while carefully checking for bounds errors on each array index. Have students write more class constructors and emphasize the importance of using the constructor to initialize the instance variables.

Question 4

What was the intent of this question?

This question assessed the students' ability to use `ArrayLists`, `Strings`, and mathematical expressions involving integers. Students also needed to understand how to use iteration and proper method calls. Students were provided with specifications for three `static` methods that would be used to create a properly spaced string of a specified length that contained a set of words given in an `ArrayList` object, `wordList`.

In part (a) students were asked to write the `totalLetters` method that sums and returns the lengths of all of the strings in its parameter `wordList`.

In part (b) students were asked to write a method to compute the minimum number of spaces necessary between each pair of words to space them out to the given length.

In part (c) students were asked to write the method that creates a single string of the given length by inserting spaces into the gaps between words so that the number of spaces in each gap is the same. If more spaces were needed to reach the given length, an additional space would be added to each gap, starting with the first gap, so that the lengths of the gaps would differ by at most 1. In doing this, they were required to call the `basicGapWidth` method written in part (b), as well as the `leftoverSpaces` method described in the problem but not implemented by the student.

To solve this problem, the student needed to traverse an `ArrayList` with no bounds errors, using the `get` method of the `ArrayList` class in an indexed `for` loop. The student could use an enhanced `for` (for-each) loop, but needed to understand the consequences of not having an index to manipulate in part (c). The student needed to properly implement a `static` method that initialized, accumulated, and returned a computed value. The student needed to properly determine the length of a string, as well as initialize an empty string and concatenate strings.

The student needed to properly call methods from within a class, including those they have implemented, and those for which they have only a description. Students needed to translate a mathematical expression into an appropriate Java expression, and understand and implement a complex algorithm from a written description.

How well did students perform on this question?

The mean score was 3.29 out of a possible 9 points with a standard deviation of 3.06.

What were common student errors or omissions?

Some students made errors in loop bounds or assumed that an enhanced `for` (for-each) loop provided an index. Students called methods with the wrong parameters or called static methods incorrectly. When computing gap size, students used the wrong formula. Students struggled with distributing both the basic gap spaces as well as the leftover spaces, particularly in dealing with the first and last words.

Students used array access instead of `ArrayList get` to access the words in `wordlist`. Students changed the value referenced by the parameter `wordList`, typically adding spaces to the strings within the list or adding additional strings of spaces between the original values.

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?

Make sure that students read the questions carefully. If specified methods are required to be used, they must be called properly and not be re-implemented.

Students should practice writing loops that accumulate a value (integer or string), remembering to initialize (not just declare) the accumulator variable and return the result at the end

Students should practice accessing arrays, strings, and `ArrayLists` and know the difference between them.

When possible, students should write the basic structure of their solution (initialize, loop, return) before filling in complex details of calculations, so that the basic solution is there in case they run out of time.

For students who have learned multiple languages, remind them of limitations of Java. For this question, many students used the `int * string` construct that Python provides but Java does not.

Students should be reminded not to change reference variables (e.g., classes and arrays) within their solution. Simply renaming the structure (e.g., `ArrayList list2 = list1`) is not sufficient to avoid this.