CollegeBoard AP

Student Performance Q&A:

2011 AP® Computer Science A Free-Response Questions

The following comments on the 2011 free-response questions for AP® Computer Science A were written by the Chief Reader, Jody Paul of Metropolitan State College in Denver. These comments provide an overview of each free-response question, explain how students performed on the question, and describe 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 array traversals, comparing array elements with a specified limit, and deleting elements from an array based on a condition. Students were given a class containing an array, `samples`, which contained sound values, and were asked to write two methods for processing those values.

The first method, `limitAmplitude`, required students to compare each element of the array to a specified limit, which was passed as a parameter. Values greater than `limit` were to be changed to `limit` and values less than `-limit` were changed to `-limit`. The method also required students to count and return the number of those changes made to the array.

The second method, `trimSilenceFromBeginning`, required students to create a new array that contained the same values in the same order as the original array but without the leading zeros. The solution required the instance variable `samples` to be updated to refer to the newly created array.

How well did students perform on this question?

This question appears to have been somewhat easier than the others. The mean score was 5.29 out of a possible 9 points, with a standard deviation of 3.19.

What were common student errors or omissions?

Most errors on this question involved confusion between array and ArrayList objects, for example, confusing element access: `a[]` *vs.* `a.get()`. Many students inappropriately attempted to use an *enhanced for* loop to change values in the array. When they intended to use a variable as a counter, many students either failed to initialize it or continually reinitialized it in the body of a

loop. Loops also ended prematurely owing to return statements appearing within the body of the loop.

Many students used the constant from the example (2000) rather than the parameter (`limit`). Students also had difficulty with declaring and creating an array; common errors were omitting the keyword `new`, omitting the element type or length of array, and declaring the array in the wrong place. Some students attempted to use string operations on arrays.

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?

- Give students extensive practice with use of the array data structure. Array and ArrayList concepts are likely to continue to constitute a substantial portion of future exams, and students need proficiency with their use.
- Students would benefit from examples of and practice with different types of loops to understand when each is appropriate and how each is used. For example, the enhanced for loop is not appropriate for use with arrays when the array elements are to be modified.
- Encourage students to use proper indentation and to include opening and closing curly braces when necessary because failing to use these can result in the student losing points (for example, a return statement appearing inside a loop).

Question 2

What was the intent of this question?

This question involved the design of a complete class within the setting of the GridWorld case study. Students were asked to design and code the `AttractiveCritter` class. An attractive critter was described as a critter that processed other actors by attempting to relocate all the other actors in the grid, including other attractive critters, one grid cell closer to itself in the direction specified by `getDirectionToward`. The question tested class definition and construction, method implementation and knowledge of the GridWorld case study. Students were instructed to write the complete class, including all instance variables and required methods. They were cautioned not to override the `act` method nor violate any postconditions of methods in the `Critter` class.

How well did students perform on this question?

This question was more difficult than others. The mean score was 3.35 out of a possible 9 points, with a standard deviation of 2.74.

What were common student errors or omissions?

The most common error was failure to guard against inappropriate self-movement (an AttractiveCritter should not be attracted to itself). Students also confused Actor and Location method calls, calling Location methods on Actor objects. They tended to test for a null Location object instead of testing for null at the specified Location in the grid.

Many students did not override `getActors`, losing points for not considering all actors in the grid. There was also confusion when working with Location objects; students incorrectly compared a Location object with null using `.equals` or used == to compare two Location objects.

Students often incorrectly changed the state of the objects in the grid by invoking `setDirection` on these objects, thereby violating a necessary postcondition. Finally, students overrode the `act` method of the Critter class although the problem specifically stated not to override `act`.

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?

- Questions that require writing an entire class should be expected, and students need proficiency with both building a class from scratch and extending an existing class.
- Remind students to read questions thoroughly and ensure that they address all aspects of the specification and requirements in their solutions.
- For case study questions, students should utilize the reference guide to verify method signatures, the nature of the values returned by those methods, and the classes in which those methods are defined. Students can also use the reference guide for sample class construction.
- Students should be mindful of the actual contents of lists returned by methods and the objects that they add to lists.
- Students need to be more familiar with using grid objects and methods used to retrieve information about the grid, about locations in the grid, and about the actors in the locations of the grid.
- Remind students to be careful not to change the state of objects when state change is not indicated.

Question 3

What was the intent of this question?

This question involved the `List` interface and two provided interfaces: `FuelTank` and `FuelRobot`. It also contained a `FuelDepot` class, which represented a fuel depot that had a number of fuel tanks arranged in a line, and a robot that moved a filling mechanism back and forth along the line so that the tanks could be filled. The `FuelDepot` class contained two important instance variables: a `FuelRobot` named `filler`, and a `List` of `FuelTanks` named `tanks`. Students needed to use these variables to call methods that were specified in `List` and the two provided interfaces. The first part of the question focused on implementing an algorithm to traverse a `List` data structure and to find a minimum. The second part focused on the state of the `filler` object. Students needed to query `filler`'s state, utilize a combination of Boolean conditions to determine whether `filler`'s state should be changed, and change `filler`'s state as appropriate. Additionally, students had to satisfy a precondition of a method that they invoked (`moveForward`).

Two unrelated `FuelDepot` methods were to be implemented. In part (a) the method `nextTankToFill` was implemented and required the use of both `filler` and `tanks` instance

variables and some of their methods as specified in the `FuelRobot` and `FuelTank` interfaces. Implementing `nextTankToFill` required an algorithm to find the minimum fuel level of any tank in the depot. This also required the use of `List` methods to access the elements of `tanks`. Finally, if the minimum fuel level was less than or equal to the value `threshold`, the index of the corresponding tank was returned. Otherwise, the index of the current tank was returned.

In part (b) the method `moveToLocation` was implemented and required the use of the `filler` instance variable. First, this method used the `moveToLocation` method's parameter and `FuelRobot` methods to determine the direction and number of spaces that `filler` must move. Then it used other `FuelRobot` methods to change direction and/or move `filler` as appropriate. It was important to be careful when calling `FuelRobot`'s `moveForward` method so that the parameter was positive. This was a `moveForward` method precondition.

How well did students perform on this question?

This question appears to have been of average difficulty. The mean score was 3.99 out of a possible 9 points, with a standard deviation of 3.15.

What were common student errors or omissions?

A common error was the attempt to use array `[]` notation to access the elements in a List. In part (a) students had difficulty identifying the object's state and behavior. Students did not use methods specified in the supplied interfaces, notably `getFuelLevel`. They tended to compare tank objects themselves rather than the fuel levels of those objects. A large number of students did not implement a correct algorithm to find the minimum. In part (b) students did not use the instance variable `filler`, which made it impossible to correctly query or maintain object state.

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?

- Give students extensive practice with use of the List collection interface. Array and List concepts are likely to continue to constitute a substantial portion of future exams, and students need proficiency with their use.
- Students should have experience with common algorithms that manipulate arrays and Lists and know the distinction between referencing array elements and List elements.
- Students should also have experience with understanding interfaces and using instance data.
- Give students practice implementing code that involves invoking methods on objects.
- Students should also have experience implementing and evaluating complex Boolean expressions as well as finding specific values, values within a range and extreme values (minimum and maximum).
- Remind students to carefully read the entire problem and note all preconditions and postconditions.

Question 4

What was the intent of this question?

This question focused on using two-dimensional arrays and utilizing abstractions. The question required students to manipulate Strings and to demonstrate understanding of how class fields interact. Students were provided with a partial class definition for the `RouteCipher` class, which indicated three relevant fields (a 2D array and the two dimensions of the array) and three methods, two of which were to be written by the student (`fillBlock` and `encryptMessage`). The third method, `encryptBlock`, was understood to be already implemented as per the given documentation and ready to be utilized.

Part (a) asked students to write the method `fillBlock`, whose task was to break apart a String into a sequence of one-character substrings and put them into a 2D array in row-major order. Nested for-loops and use of the `substring` method (of the String class) were the most common ways this was achieved.

Complications arose when the number of characters in the String did not match the size of the 2D `letterBlock`. Per the instructions, if there were extra characters in the String, they were to be ignored. If there were not enough characters in the String, the rest of the spots in the `letterBlock` were to be filled with the String `"A"` — a process called "padding."

Part (b) asked students to write the method `encryptMessage`, whose task was to take a message String and encrypt it via the `fillBlock` method written in part (a) and the `encryptBlock` method that was given to them. This required breaking the String into appropriately sized chunks that could be sent to `fillBlock`, encrypted by `encryptBlock` (which returned the encrypted message), and then combined into one String that contained the entire message in encrypted form.

How well did students perform on this question?

This question appears to have been of average difficulty. The mean score was 3.53 out of a possible 9 points, with a standard deviation of 3.22.

What were common student errors or omissions?

Many students used a position index for the substring call but did not check that index against the length of the string which resulted in potential out-of-bounds errors. Quite a few students did not attempt to pad with "A" if the string was too short.

In part (b) students often made errors in handling the last substring of the parameter. If they did partitioning via the 2-parameter substring method, they often did not address potential out-of-bounds errors on the last substring, which could be smaller than the others. Using the 1-parameter substring method was often sufficient to avoid this error.

Another common error was mishandling the empty string. (Note that calling `fillBlock` then `encryptBlock` on an empty string creates a series of "A"s, not the empty string.) It was also often assumed that `fillBlock` returned a value, which it does not.

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 need more experience with two-dimensional arrays (including traversals and manipulations), proper invocation of methods and traversal of a string via substrings. Students who attempted to traverse a string using the character datatype generally did so unsuccessfully. (Students who use constructs and classes outside of the AP Java subset must be careful to ensure that they thoroughly understand usage, especially because there is no general Java reference provided during the exam.)
- Students need more experience using the substring method, comparing strings and concatenating strings. They also need more practice with proper use of methods, parameters, instance variables and return values.
- Remind students to write solutions based on the question specification rather than specifics given in examples.