CollegeBoard AP

Student Performance Q&A:

2010 AP® Computer Science A Free-Response Questions

The following comments on the 2010 free-response questions for AP® Computer Science A were written by the Chief Reader, Jody Paul of Metropolitan State College of Denver. 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 the ArrayList data structure, element access and removal, algorithms that required processing all elements, and using instance data. Students were provided with the frameworks for two classes, `CookieOrder` and `MasterOrder`, and were asked to implement two methods in the `MasterOrder` class. In part (a) students were required to implement the method `getTotalBoxes` that returns the sum of the number of boxes of all of the cookie orders in the ArrayList instance variable. This could be accomplished by invoking `getNumBoxes` on each element of the list, accumulating and returning the sum. In part (b) students were required to implement the `removeVariety` method, which removes from the ArrayList instance variable all `CookieOrder` objects that have the same variety as the parameter, maintains an accumulator of the number of boxes removed, and returns the accumulator's final value. This could be accomplished by first invoking `getVariety` on each element of the list and performing a string comparison with the parameter. If the two strings match, the result of invoking `getNumBoxes` would be added to an accumulator and the `remove` method invoked to delete that order from the list. The accumulated total needed to be returned at the end of the method.

How well did students perform on this question?

This question appears to have been of average difficulty and comparable to questions from previous years. The mean score was 5.46 out of a possible 9 points, with a standard deviation of 3.46. Most scores were in the 6 to 9 range, but almost 19 percent of students received scores of 0 or submitted blank papers. Disregarding zeros and blank papers, the mean was 6.74.

What were common student errors or omissions?

Most errors involved confusion between array access and ArrayList collection access; that is, `[]` versus `get()`. Many students exhibited difficulty with performing element removal from

the list; they attempted to perform removal inside an enhanced for loop (for-each loop) or in an ascending-index for loop without adjusting the index. Few students attempted the most straightforward and least error-prone solution for part (b), which was to use a descending-index for loop (as shown in the canonical solution). There was a proliferation of common loop boundary errors and misuse of `==` for string comparison. Another frequent error was incorrect use of the instance data or failure to use it at all.

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 extensive practice using the ArrayList collection. Array and ArrayList concepts are likely to continue to constitute a substantial portion of future exams, and students must be proficient with their use.
- Students would benefit from examples of, and practice with, different types of loops so that they will understand when each is appropriate and how each is used. For example, the for-each loop works well in part (a) of this question but not at all well in part (b).
- It appears that students also need more experience using instance data.

Question 2

What was the intent of this question?

This question focused on the design of a complete class, including state variables (instance data), a constructor, and instance methods. Students were provided with specifications for the `APLine` class, which included private state variables capable of holding numeric values, a constructor, and `getSlope` and `isOnLine` methods. They were required to implement the entire class, including the class header. Formulas were provided for values that needed to be computed. The question tested class definition and construction, method implementation, and knowledge of how to address unwanted truncation of decimal digits that would result from integer division.

How well did students perform on this question?

This question was considered significantly easier than others on the exam and had the fewest scores of 0 and blank papers (about 10 percent of students). The mean score was 6.34 out of a possible 9 points, with a standard deviation of 3. Students generally did well, with more than half receiving scores of 8 or 9. Disregarding zeros and blank papers, the mean was 7.05.

What were common student errors or omissions?

The most common error was failing to convert, or improperly converting, values to type double prior to the division operation necessary to compute slope. A significant number of students omitted the state variables or did not declare them as private. Many students did not properly declare the class header or provide a properly formed constructor. Students appeared to have difficulty adhering to the specifications provided, especially in defining methods' return types and parameters. Another common error was confusing identifiers given in the formula specification with the actual variables used in students' code.

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 should expect questions that require writing an entire class, and they need to be proficient at building a class from scratch.
- Attention should be given to the distinction between the class header and the constructor; the roles of state variables, parameters, and local variables; and the use of method headers, their associated return types, and parameters.
- Students should practice writing classes for which the header is not given, including private instance variables, constructors, and public methods for which headers are not given. They should be reminded that sample classes exist that can be used as a guide for class construction (e.g., in the exam appendix and elsewhere on the exam).

Question 3

What was the intent of this question?

This question focused on array traversal, finding an extreme value (minimum or maximum, or both) within a specified segment of an array, and counting occurrences of array elements satisfying a given condition. Students were given a class that contained an instance data array that represented elevations and were asked to write two unrelated methods. Part (a), `isLevelTrailSegment`, required students to determine whether there was an overall difference of at most 10 in a segment of the array defined by the two parameters passed to the method. The wording of the question led students to a solution of first finding the maximum and minimum values and then checking if the difference of these values was at most 10. That approach was not required, and some students used alternatives that did not involve such computations. The second method, `isDifficult`, required students to examine the differences of pairs of consecutive array element values and determine if at least three of those differences had an absolute value greater than 30.

How well did students perform on this question?

This question appears to have been of average difficulty and slightly easier than Question 1. The mean score was 5.84 out of a possible 9 points, with a standard deviation of 3.25. Approximately 14.5 percent of students received scores of 0 or submitted blank papers. Disregarding zeros and blank papers, the mean was 6.86.

What were common student errors or omissions?

Although the question did not place any bounds on the values in the array (elevations), the example and diagram led students to assume that negative values were not possible. Because the example diagram was potentially misleading, students were not penalized for failing to consider negative values. For part (a), `isLevelTrailSegment`, the most common error was incorrectly initializing the state variable(s) used in determining a minimum or maximum. Students often mistakenly used constants other than `Integer.MIN_VALUE` and `Integer.MAX_VALUE`, or they used the element at index 0, which may not have been in the specified range. Other errors were failing to examine the element at the location given by the `end` parameter, accessing *all* elements of the array instead of just those in the specified segment, and returning prematurely from within a loop.

Students generally did very well with writing the `isDifficult` method, part (b), even if they did poorly on part (a). The most common problems were incorrectly dealing with absolute value, and bounds errors when computing the differences between consecutive 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?

- Students need to be well versed in using parameters that are passed into a method, as well as in the proper use of specified instance data.
- They also need to be aware of typical loop boundary problems associated with common coding patterns when accessing pairs of consecutive elements of an array, such as an off-by-one index at the end, an index outside the specified subset of an array, and premature return from within a loop.
- Students should also be experienced in the distinction between the use of array and ArrayList objects. Looking forward, teachers should familiarize students with the use of two-dimensional arrays.

Question 4

What was the intent of this question?

This question involved reasoning about the code from the GridWorld case study, emphasizing use of the two-dimensional grid (rather than critter or actor definitions) and ArrayList processing. Students were asked to implement two unrelated methods of a `GridChecker` class. In part (a), `actorWithMostNeighbors`, students were required to retrieve all occupied locations in a grid, determine the number of neighbors of each, and return the actor with the most neighbors or null if the grid contained no actors. This could be accomplished by calling the `getOccupiedLocations` method of the grid object, then iterating over those locations, using each as the parameter to either the `getOccupiedAdjacentLocations` or `getNeighbors` method. Some students chose to iterate over the entire grid and check each location for the presence of an actor. Students needed to create, initialize and maintain a variable to track the maximum, and to return the correct actor reference. In part (b), `getOccupiedWithinTwo`, students were required to return a list containing all occupied locations within two rows and two columns of the given parameter. There were many viable solution approaches, and the set of student solutions included most of them. Some students used the `getOccupiedLocations` method to access every occupied location in the grid, then needed to exclude those that were not within two rows and columns of the parameter. Some iterated over only locations that were within two rows and columns of the parameter, including only those in that range that were occupied. Others used the `getValidAdjacentLocations` method to access the locations within one row and one column of the parameter, then used those locations to access all occupied locations within two rows and columns of the parameter. In each case students had to ensure that each location was valid and occupied and also had to exclude the parameter from the returned list.

How well did students perform on this question?

Over one-third of students received scores of 0 or submitted blank papers, the highest number for all questions on this exam. Scores were otherwise evenly distributed, with a mean of 3.64 out of a possible 9 points and a standard deviation of 3.30. The high number of zeros and blank papers may indicate students' unfamiliarity with the particular aspect of the case study, the somewhat greater

complexity of the question, or the fact that it was the last question on the exam. Disregarding scores of 0 and blank papers, the mean was 5.49.

What were common student errors or omissions?

Many students seemed unprepared to address a case study question involving reasoning about the grid (rather than about actors). In part (a) they frequently confused the data type of the contents of lists, inappropriately mixing references to locations and actors. This was often compounded by poor choice of variable names, such as "actors" for a list of location objects. Students erroneously treated the grid as a collection — for example, by attempting to access the number of actors using a (nonexistent) size method or using the grid object in a for-each loop. When determining the maximum, they often initialized their state variables to zero under the mistaken assumption that the eventual maximum would be greater than zero, never checking for the case when all actors have no neighbors. Students also frequently omitted the null return when the grid was empty.

In part (b) errors varied depending on choice of algorithm. Students who began with all occupied locations in the grid often neglected to account for the shifting of items in the list when removing locations outside of the boundary area. Those who searched only locations within two rows and columns of the parameter often failed to check if locations were valid or occupied. In both cases students often neglected to exclude the parameter from the list. In the case where students first attempted to access locations one row and column away from the parameter, and then to access adjacent locations of those locations, they often failed to access all locations within the boundary area or removed critical contents from the list when they tried (unnecessarily) to remove duplicate locations.

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 should be reminded to read questions thoroughly and to ensure that all aspects of the specifications and requirements are addressed in their solutions.
- For case study questions, students should utilize the exam appendix to verify methods and the nature of the values returned by those methods. Students should be mindful of the actual contents of lists returned by methods and the objects that they add to those lists.
- Teachers should encourage students to use meaningful variable names to help prevent confusion of data types.
- When determining a solution, students should consider various approaches before starting implementation. Many students wrote solutions that were much longer and more complex than necessary.
- Students need to be more familiar with using grid objects and methods used to retrieve information about the grid and its contents.