Student Performance Q & A:

2016 Pilot AP® Computer Science Principles Performance Tasks

Task: Create — Applications from Ideas
Percentage of AP Score: 24%

What were students expected to demonstrate in their response to this performance task?

Students are expected to:

- design and create a program for a particular purpose. The program could be used to solve a problem, or for their own personal creativity.
- demonstrate their understanding of how to develop and implement algorithms in a program. In particular, students are expected to recognize and use algorithms as building blocks by integrating and combining them to create a new algorithm.
- use abstraction in the development of their program to manage complexity. Student will identify abstraction in their program and explain how this abstraction helped to manage the complexity of their program or how the program might function differently if this abstraction was not used.

What were some of the programs and programming languages students submitted during the 2016 Pilot Reading?

- Programs submitted for this task included, but were not limited to: choose your own adventure stories, games, simulations, mazes, apps, music, and interactive stories.
- Students used a variety of programming languages and environments to create their programs. Some of the programming languages used in the pilot reading included, but were not limited to: App Inventor, Blockly, Java, JavaScript, Python, Scratch, and Snap*!*.

How well did students address the course content related to this question? How well did students perform on the skills required on this question?

- Students were asked to create a video that demonstrates the running of at least one feature of their program. Students were able to create a video that clearly and accurately illustrated the purpose of the program.
- Students were asked to select an algorithm that integrates two or more algorithms, and accurately identify its purpose and describe how the two algorithms work independently as well as in combination. While many students were able to accurately **select** an algorithm that integrates two or more algorithms, they struggled to explain in detail what each algorithm does independently as well as in combination in the program.
- Students were asked to select an abstraction that integrated mathematical and/or logical concepts and explain how this abstraction served to manage complexity of the program. While many students were able to accurately **select** an appropriate abstraction used in their program, they struggled to write a detailed explanation of how the abstraction manages complexity in the program and why this is important in the program.

What common student misconceptions or gaps in knowledge were seen on this question?

Submission Requirement	*Common Misconceptions/Knowledge Gaps*	*Exemplary Answers*
2b: Developing a Program with a Purpose	• Referring to working with a partner or "we" throughout an entire response. This will result in a score of 0. Students are required to work independently on a significant portion of their program code, and must include a description of at least one opportunity or difficulty they encountered while programming independently. • Writing exclusively about time spent brainstorming ideas prior to beginning the design and implementation of the program.	• High scoring submissions include details about portions of program code that weren't functioning properly. The descriptions include how errors were identified and resolved independently or by working with a partner.
2c: Applying Algorithms	• Selection of algorithms that do not integrate two or more algorithms. For example, an `if..else` is not the combination of two algorithms, since `else` must follow the `if`.	• High scoring submissions include algorithms that integrate at least two different algorithms and describe what each algorithm does independently, then describe how the two algorithms work together to complete a task. Submissions can include the code segments for each algorithm individually with their description, and then the code for how the algorithms work together to create a new algorithm. *See Create Sample A.*
2d: Applying Abstraction	• Providing individual lines contained in the abstraction, without explaining the purpose of the abstraction or how it worked to manage complexity in the program.	• High scoring submissions include a comparison of how the program would function with and without the abstraction and illustrate how the abstraction works to manage complexity in the program.

Based on your experience of student responses at the AP® Reading, what advice would you offer to teachers to help them improve the performance of their students on the exam?

- Developing a Program with a Purpose – Submission Requirement 1 and 2a
  - Ensure that students have access and opportunity to practice on how to use video recording equipment prior to administering this task. **A video consisting of screen shots does not satisfy this requirement.**
  - Students need practice creating videos so that any text contained in the video is clearly visible and readable in the video.
  - Students need practice creating videos so that any sound or music included in the program can be clearly heard in the video.
  - Submissions with narration were highly effective in aiding understanding of what was shown in the video.

- Developing a Program with a Purpose – Submission Requirement 2b
  - Students are expected to write a significant portion of their program code independently. Any program code written by another person should be acknowledged by adding comments to their program code with the author's name.
  - After a student uploads their final submission, teachers will be able to review the submission to ensure appropriate acknowledgements of program code written by another person are provided. Rather than reporting this as plagiarism, teachers will be able to return the submission to the student to add the necessary acknowledgements before submitting to College Board for AP Scoring.
- Applying Algorithms and Applying Abstraction – Submission Requirement 2c and 2d
  - Students can and should include multiple program code segments to demonstrate the full utility of their algorithm and abstraction. These program code segments may come from more than one area of the program code. For example, if the algorithm calls a procedure, the student should also include the program code segment that includes the procedure in their submission.