AP Computer Science Principles Performance Task

Create — Applications from Ideas

Overview

Programming is a creative process that brings ideas to life through the development of software. Programming includes the steps of designing, implementing, testing and debugging programs. Programs can help solve problems, enable innovations, or express personal interests. For this task, the programs you develop will focus either on problem-solving or self-expression.

You will be provided with 15 hours of class time to complete and submit the following:

1. Collaborative Program Code
2. Collaborative Video
3. Individual Responses on Collaborative Work
4. Individual Program Code
5. Individual Video
6. Individual Responses on Individual Work

Your teacher will share submission guidelines that include suggestions on video and PDF tools. **Please note that audio should not be included in the video submissions;** you will provide written explanations.

General Requirements

This performance task includes both collaborative and individual components. The collaboration should include two partners or a single group of three partners if there are an odd number of students in class.

For the collaborative submissions, you are required to work with your partner(s) to:

- Design, create, and demonstrate the running of a program that solves a problem of interest to all partners and/or that represents an expression of shared personal interests among partners.
- Solicit and provide feedback.

For the individual submissions, you are required to work independently to:

- Answer questions about your collaborative program and the process of collaboration.
- Design and create a program on a topic that interests you and that solves a problem and/or provides an opportunity for self-expression.
- Answer questions about your individual program.

Program Requirements

The collaborative program must make active use of all partners. For example, you may choose to employ *Pair Programming*, in which one partner “drives” (types and uses the mouse) while the other “navigates” (reviews and helps to guide what the driver is doing), with the partners changing roles every 20 minutes. Another method of collaboration is for each partner to develop pieces of the program, combine those pieces, and provide frequent feedback to each other during the development process. Other methods of collaboration are permitted as long as the method of collaboration requires and makes active use of all partners.

You will be required to submit an explanation of the development process used when creating your collaborative program. You may find it useful to keep a log or journal of your program progression as you go through this collaborative program development process.

The program you produce individually must be different from the one you write collaboratively and from your partner’s individually-produced program. You can choose to add features to the collaboratively-produced program or create a new program. You can write the individual program either in the same language or in a different language from the collaborative program. You will be required to answer questions about abstractions and algorithms used in your individual program.

Both programs must demonstrate:

- The richness of the programming language and the basic programming elements of the language. Programs should demonstrate appropriate use of numbers, text, variables, statements, mathematical expressions with arithmetic operators, logical and Boolean operators and expressions, decision statements, iteration, and collections such as lists.
- The creation of abstractions to develop and manage the complexity of the program (e.g., functions/procedures, parameterization, or data abstractions).
- The use of algorithms (including sequencing, selection, and/or iteration) as building blocks for the program.

Collaborative Program Submissions

You and your partner(s) will submit the following components. Either partner can submit the collaborative program code and video; both will see the submissions in the system and receive credit.

- Collaborative Program Code
- Collaborative Video

Each partner will independently submit the following component:

- Individual Responses on Collaborative Work

1. Collaborative Program Code

Submit one PDF document per set of partners that includes program source code, either as text or as screenshots. Include comments that help the reader better understand your code.

2. Collaborative Video

Submit one video per set of partners in mp4, wmv, avi, or mov format (1 minute maximum length and 30 mb maximum size) that displays the successful running of the program and illustrates two primary features of the program.

3. Individual Responses on Collaborative Work

Working on your own, each student will submit one PDF document. Respond directly to each prompt and clearly label your responses (a, b, c, d, e, f). **You have a maximum of 200 words for each response**. Cite specific evidence in responding to prompts c, d, e, f.

a. Explain what your collaborative video illustrates.
b. Identify the programming language and purpose of your collaborative program. Describe any required input from the person running the program and the output produced by the program.
c. Describe how collaboration facilitated the selection of a focus for your program that was meaningful to both you and your partner.
d. Describe how collaboration changed the size or complexity of your program.
e. Describe how collaboration made it easier to find and correct errors in your program.
f. Describe the incremental development of the program. Describe two points of the development process—one point at roughly one-third of the way into the programming process and one point during the last third of the program's development. Discuss how your program functioned, and describe the difficulties encountered at each of the two points.

Individual Program Submissions

You will submit the following components for your individual program:

- Individual Program Code
- Individual Program Video
- Individual Responses on Individual Work

4. Individual Program Code

Each student will submit one PDF document that includes program source code, either as text or as screenshots.

- Include comments that help the reader better understand your code.
- Demonstrate your program implements an algorithm that is essential for your program by drawing a **red rectangle** around the segment of code that illustrates your algorithm.
- Demonstrate that your program uses abstraction by drawing a **blue rectangle** around the segment of code that illustrates abstraction.

5. Individual Video

Each student will submit one video in mp4, wmv, avi, or mov format (1 minute maximum length and 30 mb maximum size) that displays the successful running of the program and illustrates two primary features of the program.

6. Individual Responses on Individual Work

Each student will submit one PDF document. Respond directly to each prompt and clearly label your responses (a, b, c, d). **You have a maximum of 200 words for each response.**

a. Explain what your video illustrates.
b. Identify the programming language and describe how your program solves a problem or represents self-expression. Explain your rationale for selecting the focus of your program.
c. Describe what the algorithm you chose accomplishes (red rectangle in program code document).
d. Explain the abstraction you selected (blue rectangle in program code document).

Learning Objectives Assessed

This Performance Task addresses the following AP Computer Science Principles Learning Objectives:

1.2.1 Create a computational artifact for creative expression. [P2]*****

1.2.2 Create a computational artifact using computing tools and techniques to solve a problem. [P2]******

1.2.3 Create a new computational artifact by combining or modifying existing artifacts. [P2]

1.2.4 Collaborate in the creation of computational artifacts. [P6]

2.2.1 Develop an abstraction when writing a program or creating other computational artifacts. [P2]

2.2.2 Use multiple levels of abstraction to write programs. [P3]

4.1.1 Develop an algorithm for implementation in a program. [P2]

4.1.2 Express an algorithm in a language. [P5]

5.1.1 Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge. [P2]

5.1.2 Develop a correct program to solve problems. [P2]**

5.1.3 Collaborate to develop a program. [P6]

5.2.1 Explain how programs implement algorithms. [P3]

5.3.1 Use abstraction to manage complexity in programs. [P3]

5.4.1 Evaluate the correctness of a program. [P4]

5.5.1 Employ appropriate mathematical and logical concepts in programming. [P1]

*****Learning Objective 1.2.1 is assessed if students decide to create a program for the purpose of self-expression.

******Learning Objective 1.2.2 is assessed if students decide to create a program to solve a problem.