2025

AP® Computer Science Principles

Scoring Guidelines

Set 1

Video, Program Requirements, and Written Response 1 — 3 points

General Scoring Notes

- Written responses should be evaluated solely on the rationale provided.
- Responses must demonstrate all scoring criteria, including those within bulleted lists, in each reporting category to earn the point for that category**.**
- Terms and phrases defined in the terminology list are italicized when they first appear.

Reporting Category	Scoring Criteria	Decision Rules
Course Project: Video **(0–1 points)**	The video demonstrates the running of the program including: • *input* • *program functionality* • *output*	**Consider the video (or Program Code if necessary) when scoring this point.** • The video needs to show at least one example of the program's functionality. • If the source of the input is unclear from the video, consider the full program code file when scoring this point. **Do NOT award a point if the following is true:** • The video does not show a demonstration of the program running (screenshots or storyboards are not acceptable and would not be credited).
Course Project: Program Requirements **(0–1 points)**	The program code includes: • A *student-developed procedure* • A call to the student-developed procedure • A *list* (or other *collection type*) • A use of the list • *Selection* • *Iteration*	**Consider the Personalized Project Reference (or Program Code if necessary) when scoring this point.** • If the program requirements do not appear in the Personalized Project Reference, consider the full program code file when scoring this point. • The procedure does not need to have a *parameter* to earn this point. • The *code segments* demonstrating selection and iteration do not need to appear in the same *algorithm* to earn this point. • The code segments demonstrating selection and iteration do not need to be contained in a procedure to earn this point. **Do NOT award a point if any one or more of the following is true:** • The list is a one-element list. • The use of the list is irrelevant (i.e., not connected to the program's functionality). • The call to the procedure is inconsistent with the procedure header (unless allowed by the programming language). • The use of either the selection or the iteration is trivial (i.e., does not affect the outcome of the program).

Reporting Category	Scoring Criteria	Decision Rules
Written Response 1: Program Design, Function, and Purpose **(0–1 points)**	The written response: • identifies at least one example output of the program. • explains how the identified output(s) show an aspect of the program's functionality.	**Consider the Video and Written Response 1 when scoring this point.** • If the video is not available or does not show the described output, consider the full program code file when scoring this point. • Either a specific example of the output or a general description of the output can meet this criteria. • The response may describe more than one output but only needs to explain how at least one of the outputs shows an aspect of the program's functionality. If the response describes more than one output, all descriptions given must be correct. **Do NOT award a point if any one or more of the following is true:** • The identification of the example output(s) are implausible, inaccurate, or inconsistent with the program. • The explanation of how the output(s) show an aspect of the program's functionality is implausible, inaccurate, or inconsistent with the program. • The output does not arise from the normal execution of the program (e.g., an error message produced by a compiler).

Written Response 2 3 points

General Scoring Notes

- Written responses should be evaluated solely on the rationale provided.
- Written responses must demonstrate all scoring criteria, including those within bulleted lists, in each reporting category to earn the point for that category**.**
- Terms and phrases defined in the terminology list are italicized when they first appear.

Reporting Category	Scoring Criteria	Decision Rules
Written Response 2a: Algorithm Development **(0–1 points)**	The written response: • identifies the Boolean expression in the first selection statement. • identifies a specific value or set of values that will cause the Boolean expression of the selection statement to evaluate to `true`. • explains why the specified value or set of values will cause the expression to evaluate to `true`.	**Consider the Personalized Project Reference and Written Response 2(a) when scoring this point.** • If multiple selection statements are included in the Procedure section of the Personalized Project Reference, use the first selection statement to determine whether the point is earned. • The selection statement does not need to be contained in a procedure to earn this point. • The response does not have to explicitly state the Boolean expression as long as it is described. • The identified Boolean expression can include the header of the selection statement (e.g., `if (x > 2)` instead of `x > 2`). • The response may earn this point for a selection statement that either does or does not contain an `else` clause. **Do NOT award a point if any one or more of the following is true:** • The Procedure section of the Personalized Project Reference does not contain a selection statement. • The identified Boolean expression does not match the code in the first selection statement. • The identified value or set of values that will cause the expression to evaluate to `true` does not match the code in the first selection statement. • The response identifies a Boolean expression or value (or set of values) that is implausible, inaccurate, or inconsistent with the program.

Reporting Category	Scoring Criteria	Decision Rules
Written Response 2b: Errors and Testing **(0–1 points)**	The written response: • describes a modification another programmer could make that would cause the *procedure* to have a logic error. • describes how the behavior of the procedure would change because of this introduced error.	**Consider the Personalized Project Reference and Written Response 2(b) when scoring this point.** • The modification can be changing code, adding code, and/or deleting code. • If multiple procedures are included in part (i) of the Procedure section of the Personalized Project Reference: • Use the procedure referenced in the written response to determine whether the point is earned. • If no procedure is referenced in the written response, then use the first procedure to determine whether the point is earned. • A logic error is a mistake in an algorithm or program that causes it to behave incorrectly or unexpectedly. Responses that describe a modification that would lead to the program crashing may earn this point. • The response may describe more than one modification that leads to a logic error. If the response describes more than one modification, the change in behavior must be described correctly for every modification. **Do NOT award a point if any one or more of the following is true:** • A procedure is not included in part (i) of the Procedure section of the Personalized Project Reference. • The response does not apply to the procedure in part (i) of the Procedure section of the Personalized Project Reference. • The response states only that the program will not run without further explanation. • The response describes a modification or change in behavior that is implausible, inaccurate, or inconsistent with the procedure or the described modification.

Written Response 2c: Data and Procedural Abstraction **(0–1 points)**	The written response: • explains how the *code segment* that accesses or manipulates *data stored in the list* (or other *collection type*) would need to change if another programmer added several new elements to the end of the list. OR • explains why no changes to the code segment would be necessary.	**Consider the Personalized Project Reference and Written Response 2(c) when scoring this point.** • If multiple lists are included in the List section of the Personalized Project Reference: • Use the list(s) referenced in the written response to determine whether the point is earned. • If no list is referenced in the response, then use the first list to determine whether the point is earned. • The response may earn this point if the described code modifications change the functionality of the program. **Do NOT award a point if any one or more of the following is true:** • A list (or other collection type) is not included in part (i) of the List section of the Personalized Project Reference. • The response does not apply to the list included in the List section of the Personalized Project Reference. • The use of the list is trivial and does not assist in fulfilling the program’s purpose. • The response describes a modification that makes the list unnecessary. • The response includes an explanation that is implausible, inaccurate, or inconsistent with the program.

AP Computer Science Principles Create Performance Task Terminology

Algorithm: An algorithm is a finite set of instructions that accomplish a specific task. Every algorithm can be constructed using combinations of sequencing, selection, and iteration.

Arguments: The values of the parameters when a procedure is called.

Code segment: A code segment refers to a collection of program statements that are part of a program. For text-based, the collection of program statements should be continuous and within the same procedure. For block-based, the collection of program statements should be contained in the same starter block or what is referred to as a "Hat" block.

Collection type: Aggregates elements in a single structure. Some examples include: databases, hash tables, dictionaries, sets, or any other type that aggregates elements in a single structure.

Data stored in a list: Input into the list can be through an initialization or through some computation on other variables or list elements.

Input: Program input is data that are sent to a computer for processing by a program. Input can come in a variety of forms, such as tactile (through touch), audible, visual, or text. An event is associated with an action and supplies input data to a program.

Iteration: Iteration is a repetitive portion of an algorithm. Iteration repeats until a given condition is met or for a specified number of times. The use of recursion is a form of iteration.

List: A list is an ordered sequence of elements. The use of lists allows multiple related items to be represented using a single variable. Lists are referred to by different terms, such as arrays or arraylists, depending on the programming language.

List being used: Using a list means the program is creating new data from existing data or accessing multiple elements in the list.

Output: Program output is any data that are sent from a program to a device. Program output can come in a variety of forms, such as tactile, audible, visual, movement, or text.

Parameter: A parameter is an input variable of a procedure. Explicit parameters are defined in the procedure header. Implicit parameters are those that are assigned in anticipation of a call to the procedure. For example, an implicit parameter can be set through interaction with a graphical user interface.

Procedure: A procedure is a named group of programming instructions that may have parameters and return values. Procedures are referred to by different names, such as method, function, or constructor, depending on the programming language. A procedure is executed through the use of a procedure call.

Program functionality: The behavior of a program during execution, often described by how a user interacts with it.

Purpose: The problem being solved or creative interest being pursued through the program.

Selection / conditional statement: A selection / conditional statement affects the sequential flow of control by executing different statements based on a condition being true or false. The use of if-statements and try / exception statements are examples of selection / conditional statements.

Sequencing: The application of each step of an algorithm in the order in which the code statements are given.

Student-developed procedure / algorithm: Program code that is student-developed has been written (individually or collaboratively) by the student who submitted the response. Calls to existing program code or libraries can be included but are not considered student-developed. Event handlers are built-in abstractions in some languages and will therefore not be considered student-developed. In some block-based programming languages, event handlers begin with "when".