2022

AP® CollegeBoard

AP® Computer Science Principles

Sample Student Responses and Scoring Commentary

Inside:

Performance Task—Create

☑ Scoring Guidelines

☑ Scoring Commentary

Student Samples provided separately

Create Performance Task — 6 points

General Scoring Notes

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

Reporting Category	Scoring Criteria	Decision Rules
Row 1 **Program Purpose and Function** **(0–1 points)**	The video demonstrates the running of the program including: • *input* • *program functionality* • *output* **AND** The written response: • describes the overall *purpose* of the program. • describes what functionality of the program is demonstrated in the video. • describes the input and output of the program demonstrated in the video.	**Consider ONLY the video and written response 3a 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). • The described purpose is actually the functionality of the program. The purpose must address the problem being solved or creative interest being pursued through the program. The function is the behavior of a program during execution and is often described by how a user interacts with it.
Row 2 **Data Abstraction** **(0–1 points)**	The written response: • includes two *program code segments*: - one that shows how *data has been stored in this list* (or other *collection type*). - one that shows the data in this same *list being used* as part of fulfilling the program's purpose. • identifies the name of the variable representing the list being used in this response. • describes what the data contained in this list is representing in the program.	**Consider ONLY written response 3b when scoring this point.** **Requirements for program code segments:** • The written response must include two clearly distinguishable program code segments, but these segments may be disjointed code segments or two parts of a contiguous code segment. • If the written response includes more than two code segments, use the first two code segments to determine whether or not the point is earned. **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 does not assist in fulfilling the program's purpose.

Reporting Category	Scoring Criteria	Decision Rules
Row 3 Managing Complexity **(0–1 points)**	The written response: • includes a program code segment that shows a list being used to manage complexity in the program. • explains how the named, selected list manages complexity in the program code by explaining why the program code could not be written, or how it would be written differently, without using this list.	**Consider ONLY written response 3b when scoring this point.** **Responses that do not earn the point in row 2 may still earn the point in this row.** **Do NOT award a point if any one or more of the following is true:** • The code segments containing the lists are not separately included in the written response section (not included at all, or the entire program is selected without explicitly identifying the code segments containing the list). • The written response does not name the selected list (or other collection type). • The use of the list is irrelevant or not used in the program. • The explanation does not apply to the selected list. • The explanation of how the list manages complexity is implausible, inaccurate, or inconsistent with the program. • The solution without the list is implausible, inaccurate, or inconsistent with the program. • The use of the list does not result in a program that is easier to develop, meaning alternatives presented are equally complex or potentially easier. • The use of the list does not result in a program that is easier to maintain, meaning that future changes to the size of the list would cause significant modifications to the code.
Row 4 Procedural Abstraction **(0–1 points)**	The written response: • includes two program code segments: - one showing a *student-developed procedure* with at least one *parameter* that has an effect on the functionality of the procedure. - one showing where the student-developed procedure is being called. • describes what the identified procedure does and how it contributes to the overall functionality of the program.	**Consider ONLY written response 3c when scoring this point.** **Requirements for program code segments:** • The procedure must be student developed, but could be developed collaboratively with a partner. • If multiple procedures are included and none are specifically called out in the written response, use the first procedure listed to determine whether the point is earned. • The parameter(s) used in the procedure must be explicit. Explicit parameters are defined in the header of the procedure. **Do NOT award a point if any one or more of the following is true:** • The code segment consisting of the procedure is not included in the written responses section. • The procedure is a built-in or existing procedure or language structure, such as an event handler or main method, where the student only implements the body of the procedure rather than defining the name, return type (if applicable), and parameters. • The written response describes what the procedure does independently without relating it to the overall function of the program.

Reporting Category	Scoring Criteria	Decision Rules
Row 5 **Algorithm Implementation** **(0–1 points)**	The written response: • includes a program code segment of a *student-developed algorithm* that includes - *sequencing* - *selection* - *iteration* • explains in detailed steps how the identified algorithm works in enough detail that someone else could recreate it.	**Consider ONLY written response 3c when scoring this point.** **Responses that do not earn the point in row 4 may still earn the point in this row.** **Requirements for program code segments:** • The algorithm being described can utilize existing language functionality or library calls. • An algorithm that contains selection and iteration also contains sequencing. • An algorithm containing sequencing, selection, and iteration that is not contained in a procedure can earn this point. • Use the first code segment, as well as any included code for procedures called within this first code segment, to determine whether the point is earned. • If this code segment calls other student-developed procedures, the procedures called from within the identified procedure can be considered when evaluating whether the elements of sequencing, selection, and iteration are present as long as the code for the called procedures is included. **Do NOT award a point if any one or more of the following is true:** • The response only describes what the selected algorithm does without explaining how it does it. • The description of the algorithm does not match the included program code. • The code segment consisting of the selected algorithm is not included in the written response. • The algorithm is not explicitly identified (i.e., the entire program is selected as an algorithm without explicitly identifying the code segment containing the algorithm). • The use of either the selection or the iteration is trivial and does not affect the outcome of the program.

Reporting Category	Scoring Criteria	Decision Rules
Row 6 **Testing** **(0–1 points)**	The written response: • describes two calls to the selected procedure identified in written response 3c. Each call must pass a different *argument(s)* that causes a different segment of code in the algorithm to execute. • describes the condition(s) being tested by each call to the procedure. • identifies the result of each call.	**Consider ONLY the written response for 3d and the selected procedure identified in written response 3c.** **Responses that do not earn the point in row 4 may still earn the point in this row.** **Requirements for program code segments:** • Consider implicit or explicit parameters used by the selected procedure when determining whether this point is earned. 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. • A condition that uses the procedure's parameter(s) to execute two different code segments can earn this point. • A condition that uses the procedure's parameter(s) to execute or bypass a code segment can earn this point. **Do NOT award a point if any one or more of the following is true:** • A procedure is not identified in written response 3c. • The written response for 3d does not apply to the procedure in 3c. • The two calls cause the same exact sequence of code in the algorithm to execute even if the result is different. • The response describes conditions being tested that are implausible, inaccurate, or inconsistent with the program. • The identified results of either call are implausible, inaccurate, or inconsistent with the program.

AP Computer Science Principles Create Performance Task Terminology (in order of appearance in the scoring guidelines)

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.

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

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.

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

Program 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.

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.

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

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.

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

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."

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.

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.

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.

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

Selection: Selection determines which parts of an algorithm are executed based on a condition being true or false. The use of try / exception statements is a form of selection statements.

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.

Argument(s): The value(s) of the parameter(s) when a procedure is called.

Create Performance Task

Note: Student samples are quoted verbatim and may contain spelling and grammatical errors.

Overview

The Create Performance Task is designed to give students an opportunity to develop a program that solves a problem for the user or allows the pursuit of a creative interest. Students should be able to demonstrate the program running in a short video and explain its purpose, how it functions, and how it handles input and output of information as shown in the video.

Programs typically process collections of data to help the user gain insight and make decisions. This task also requires students to demonstrate their understanding of data abstraction, using at least one list (or equivalent collection type) to hold data that is critical to fulfilling the program's purpose. Students must explain how the list manages complexity in the program, by either explaining why their program could not function without the list or why their program would require a more complex implementation without the use of the list, to demonstrate the importance of using this abstraction when processing larger amounts of data.

Programs use procedures to break a larger computational task into smaller subtasks to make a program easier to develop and test. This task also requires students to use procedural abstraction to write a procedure with at least one explicit parameter that demonstrates the use of sequencing, selection, and iteration, along with a call to this procedure. The student should be able to explain how the procedure works in detail, what the procedure does in summary, and how the procedure contributes to the overall functionality of the program. Finally, the student should be able to explain how to test the procedure for correctness using its parameter(s), using two examples that cause different behavior and results to occur.

Create Performance Task (continued)

Sample Identifier: A

Score:
Row 1: 1
Row 2: 1
Row 3: 1
Row 4: 1
Row 5: 1
Row 6: 1

Row 1:
The response earned the point for this row, meeting all six criteria:

- The video demonstrates the running of the program, showing input into the program through the user placing tiles on a grid. Output includes the display of the game simulation. It also shows the user pausing, editing, and replaying the simulation. This satisfies the first three criteria for the video.
- The response specifies the program's purpose is "to serve as entertainment for a user."
- The response describes the functionality demonstrated in the video. The response states, "The program allows a user to place and remove tiles on a grid, which are then progressed programatticaly by the rules of Conway's Game of Life, with the program updating the screen after every generation. The user is also at all times allowed to pause, restart, and reset the program, editing and replaying the simulation."
- The response describes the input and the output. Regarding input, the response states, "the keys `space`, `p`, and `r`, as well as a user's mouse clicks/position. The user's mouse position and clicking are taken in." Regarding output, the response states, "an output as a placed or removed cell on a grid, followed by the progression of those cells through simulated generations."

Row 2:
The response earned the point for this row, meeting all three criteria:

- Two distinct code segments are provided. The first segment shows data being stored, and data being accessed, in the identified list, `startGrid`. The second segment is not needed to score the response.
- The name of the list is identified as `startGrid`. The response identifies other lists, but they are not considered in the scoring.
- The response identifies what is stored in the list. The response states, "Each item in these lists stores the status of one of the 10,800(120 x 90) possible cells on screen. A `0` being dead, and a `1` being alive. From the position in the list can be derived the coordinates of a cell, and vice versa."

Create Performance Task (continued)

Row 3:
The response earned the point for this row, meeting both of the criteria:

- The response includes code that uses a list to manage complexity by accessing and storing the status (1 or 0) of 10,800 items.
- The response explains how the code could not be written without a list. The response states, "10,800 items must be kept track of, with a single position corresponding to a single cell, and a way to find one from the other. To do so with variables or some other method would be a monumental task, and very inefficient, so the simplest and easiest way is to do so is with a list. A list also consolidates all that information in one place, making it easier to use and keep track of. In a list, it is also easier to replace one individual item than it would be to replace a single character in a variable or other tracking method."

Row 4:
The response earned the point for this row, meeting both of the criteria:

- The response includes a student-developed procedure, `replaceList`, with two parameters, `current` and `replacement`, that are used in the procedure. The response also includes a call to this procedure.
- The response describes the functionality of the procedure. The response states, "This procedure takes as inputs identifying numbers for two lists. The first will be copied over to replace the second." The response describes how this procedure contributes to the overall functionality of the program. The response states, "It is used in the program to store all cells in a generation, replace a different list with those same cells, and be able to recall a previous set of points through the copying of lists."

Row 5:
The response earned the point for this row, meeting both of the criteria:

- The student-developed algorithm within procedure, `replaceList`, includes sequencing, selection (if statement), and iteration (repeat loop).
- The response explains in detail how the algorithm in the procedure works so it can be recreated. The response states, "The algorithm uses nested if-else statements to identify which list will be copied from, by looking at the first parameter. Inside the if statement matching that parameter, are another set of if statements to determine where the list should be copied over to. This could be adapted to any number of lists, but is in this case just three, meaning two if statements within each of three nested if-else, as of course a list cannot be copied to itself." Additional explanation is given.

Create Performance Task (continued)

Row 6:
The response earned the point for this row, meeting all of the three criteria:

- The response describes two different calls to the specific procedure to result in different code being executed. The response states, “This call takes inputs of `0, 1` being the current and replacement parameters respectively.” The response states that the second call is “parameter inputs of `2, 1,` being the current and replacement parameters respectively.”
- The response describes the conditions being tested. The response states, “This call checks which list is being copied from, and where to. With parameter `'current'` being `0`, the first if will be entered, where, with parameter `'replacement'` as `1`, the first if will again be entered.” The response describes the second condition, “As parameter `'current'` is `2`, and `'replacement'` is `1`, the first if will fail, as will the second, and the third entered. Within that, the first if fails again, and the second is entered.”
- The response describes the results of the two calls, leading to two different results. The response states the first result as, “The entirety of the list `'startGrid'` will be copied over to replace list `'currentGrid'`,” and the result of the second call as “The entirety of the list `'nextGrid'` will be copied over to replace list `'currentGrid'`."

Create Performance Task (continued)

Sample Identifier: B

Score:
Row 1: 1
Row 2: 1
Row 3: 1
Row 4: 1
Row 5: 1
Row 6: 1

Row 1:
The response earned the point for this row. The response met all six criteria:

- The video demonstrates the running of the program, showing input into the program through keyboard typing. Output includes the prompts for the word input and the poems being displayed. This satisfies the first three criteria for the video.
- The response specifies the program's purpose as creative expression, "to explore the user's creativity through the expression of poetry using their words and images as well as preferences of formatting in order to generate some unique poems."
- The response describes the functionality demonstrated in the video. The response states, "The video shows the generation of a poem as specified by the user's inputs of words like pine, star, lies, etc., and the setting of with articles. It results in multiple different poems being generated with different word orders (3, as specified by the user). Then another generation occurs with different words (forest, expanse, glows, etc.) with no articles and 5 poems being generated."
- The response describes the input and the output. Regarding input, the response states, "The input of both typing words to insert into the poem and a typing of yes or no to set articles or no articles as well as number of desired poems." Regarding output, the response states that the program "generated all result in the output of a certain specified number of poems according to those words and specifications."

Row 2:
The response earned the point for this row, meeting all three criteria:

- Two distinct code segments are provided, the first showing data being stored in the identified list and the second showing the data being accessed from the identified list under a new name since it is passed as a parameter.
- The name of the list is identified as `nounList`, although it is also referred to as `wordList` when it is passed into the `selectWord` procedure.
- The response identifies what is stored in the list. The response states, "The data in the list contains the nouns that must be in certain specific locations within the poem (so that the poem makes logical sense)."

Create Performance Task (continued)

Row 3:
The response earned the point for this row, meeting both of the two criteria.

- The response includes code that uses lists of words to manage complexity when generating poetry.
- The response explains how the code would be written differently without the list. The response states, "If I did not have this list, I would have to store the noun inputs in different variables and keep invoking those different variables instead, as well as finding another way to randomize selection, perhaps using the generation of a random integer, each corresponding to its own selection/if statement, but in that case I would have to write individual statements for each possible int generated and make the code a lot more cluttered."

Row 4:
The response earned the point for this row, meeting the two criteria:

- The response includes a student-developed procedure, `createPoems`, with four parameters that are used in the procedure.
- The response describes the functionality of the procedure as it states that it is "putting together the final selection of poems." The response describes how the procedure contributes to the overall program by stating that it is used in conjunction with the user inputs. The response states that this procedure builds poems "according to how many the user specified and whether the user wanted articles in the poems."

Row 5:
The response earned the point for this row, meeting both of the two criteria:

- The student-developed algorithm within procedure, `createPoems`, includes sequencing, selection (if statement), and iteration (while loop).
- The response explains in detail how the algorithm in the procedure works so it can be recreated. The response states that it "sequentially first defines a variable to count the number of poems generated and defines an empty string, then has iteration that generates as many poems as the user specified by running through the selection statements which call `articlePoem` or `noArticlePoem` each time. The selection if statements between `articlePoem` or `noArticlePoem` respectively call either the procedure for a poem with articles or one without depending on the user's input (not pictured) and thus the argument for `aSetting`, and whichever type of poem specified earlier will be added to the string. Finally, the string is returned."

Create Performance Task (continued)

Row 6:
The response earned the point for this row, meeting all three of the three criteria:

- The response describes two calls to the specific procedure to result in different code being executed, varying the argument for the last parameter. The first call is `createPoems(nounList, verbList, adverbList, 1)`, and the second call is `createPoems(nounList, verbList, adverbList, 0)`. The response states that the last parameter represents "differing settings for `articlePreference`."
- The response describes the condition being tested by stating "The first call checks the `aSetting` (article setting) and the if statement on line 37 which checks if `aSetting` equals `1` will then execute line 38." The response also states, "The second call checks the `aSetting` (article setting) except since the argument equals `0` fails the check of the if statement on line 37 and instead passes the check on the if statement of line 39 to execute line 40." The response also states that the code is testing "to generate poems either with or without articles based on if the user input yes or no."
- The response specifies the different results of the two procedure calls. The response states in the first case that "the final output poems will have "`The`" and "`And`" in the content." The responses states in the second case that "the final output poems will not have "`The`" and "`And`" in the content."

Create Performance Task (continued)

Sample Identifier: C

Score:
Row 1: 1
Row 2: 1
Row 3: 1
Row 4: 1
Row 5: 1
Row 6: 1

Row 1:
The response earned the point for this row, meeting all of the six criteria:

- The video demonstrates the running of the program, showing input into the program through a text field to enter letters for the puzzle. Output includes the display of letters or the number of lives reducing for wrong guesses, along with the final screen to show if the player wins or not. This satisfies the first three criteria for the video.
- The response specifies the program's purpose is to "help you recognize new words and expand your vocabulary."
- The response describes the functionality demonstrated in the video. The response states, "In the video it shows the user inputting many letters and the program either outputting the letter where the letter belongs in the randomly generated word or taking a life from the user since the letter was not found in the word."
- The response describes the input and the output. Regarding input, the response states, "The video shows the user inputting the letter `"p"` and "The user also inputs numbers, symbols, multiple letters in a single guess." Regarding output, the response states, "the program outputting the letter in the spot that it belongs in the word" and "prompted with a winning screen."

Row 2:
The response earned the point for this row, meeting all three criteria:

- Two distinct code segments are provided. The first segment shows data being stored in the identified list, and the second segment shows data being accessed from the identified list in a for loop.
- The name of the list is identified as `letOfGuessWord`.
- The response identifies what is stored in the list. The response states that the list "holds the individual letters of the word that the user is trying to guess."

Create Performance Task (continued)

Row 3:
The response earned the point for this row, meeting both of the two criteria.

- The response includes code that uses a list to manage complexity by storing the letters of the word to be guessed. The code scans the list based on its length so a longer word could be implemented.
- The response explains how the code would be written differently without the list. The response states, "If the program were to run without the `"letOfGuessWord"` list, it would make everything very inefficient because I would have to make 5 new variables to store the individual letters, that is taken from the randomly generated word, into each variable. This will also make my for loop more complicated since I would have to loop trough the strings to see if the letter that the user inputted, is in fact in the word."

Row 4:
The response earned the point for this row, meeting both of the criteria:

- The response includes a student-developed procedure, `guessWords`, with a parameter that is used in the procedure. The response also includes a call to this procedure.
- The response describes the functionality of the procedure. The response states, "If the user has gotten it correct, the letter will pop up in the space where it belongs. If the program does not find the letter in the word, it will take a life away from the user." The response describes how this procedure contributes to the overall functionality of the program. The response states, "The function does most the work by basically comparing the letters of the randomly generated word to the letter that the user has inputted."

Row 5:
The response earned the point for this row, meeting both of the two criteria:

- The student-developed algorithm within procedure, `guessWords`, includes sequencing, selection (if statement), and iteration (for loop).
- The response explains in detail how the algorithm in the procedure works so it can be recreated. The response states, "I have a Boolean, `"check"`, that is set to false, a variable `"count"` that is set to `0`, and a variable `"lives"` that is set to `6`. Then I use a for loop that loops through the entire list by the length of the word that the user is trying to guess. Inside the for loop I have an if statement that compares the letter that the user inputted, with all of the letters in the randomly generated word. If the if statement condition is true it does the following: it sets the letter that the user inputted to let `x,x` being the index of where the letter was found in the randomly generated word, then it will change `"check"` to true since it did indeed find a letter in the word, lastly it will add `1` to `"count"` since it needs to keep track of how many times the user has guessed a letter correctly." Additional explanation is given.

Create Performance Task (continued)

Row 6:
The response earned the point for this row, meeting all of the three criteria:

- The response describes two different calls to the specific procedure to result in different code being executed. The response states that "the user inputs the number `"1"` and inputs the letter `"h"` when the word to be guessed is `"hello"`. The response uses the word `"inputs"` to indicate the setting of the parameter, `letter`.
- The response describes the conditions being tested. The response states, "the number `"1"` not being found in `"hello"`," and "the letter `"h"` being found in `"hello"`." Note that the information provided in the second part of the response is not relevant to the answer.
- The response describes the results of the two calls, leading to two different results. The response states, "The result of the first call is false," and "The result of the second call is true."

Create Performance Task (continued)

Sample Identifier: D

Score:
Row 1: 0
Row 2: 1
Row 3: 1
Row 4: 1
Row 5: 1
Row 6: 1

Row 1:
The response did not earn the point for this row, meeting five of the six criteria:

- The video demonstrates the running of the program, showing input into the program using the drop-down menus to select two characters for the battle, and output consisting of a display of the values for the various rankings of each character and the winner of the battle using an image. This satisfies the first three criteria for the video.
- The response describes the function of the program but not its purpose. The response states, "The program aims to determine the winner of a hypothetical fight between two Marvel characters, based on which character has the higher average in six categories." This did not satisfy a criteria for this row.
- The response describes the functionality demonstrated in the video. The response states, "This video demonstrates the use of buttons to change screens, how the user selects characters from the drop-down box, the display changing based on the character selected, and the winner screen's outputs based on the winner."
- The response describes the input and the output. For input, the response states that "the different button clicks to change the screens and the character names selected from each drop-down box." For output, the response states that "the different screens, the individual rankings of the two characters, an image of both characters on the character screen, and the image of the winner (or a tie image) and that character's name (or 'tie')."

Row 2:
The response earned the point for this row, meeting all three criteria:

- Two code segments are provided. The first segment shows data being stored in the identified list, and a second segment shows data being accessed from the list in a loop via a parameter that is set to the identified list.
- The list is identified in the response as `firstCharacterList`.
- The response identifies what is stored in the list. The response states, "The data within the list represents the power rankings and URL for the image of the character that was selected from the first drop-down box."

Create Performance Task (continued)

Row 3:
The response earned the point for this row, meeting both criteria:

- The response includes a list that combines six rankings and an image URL for a single character into one collection to pass to the function that determines a winner, managing complexity in the program code.
- The response provides an accurate explanation about what would happen if the list were not used. The response states that the list "allows the information for the character selected to be stored in one variable instead of seven variables which helps to condense the code needed to accomplish the goal of this program. If the list was not used more parameters would be needed for the `findWinner()` function and each variable would need to be added together for the first character selected before being divided by 6 to get the average."

Row 4:
The response earned the point for this row, meeting both criteria:

- The response includes a student-developed procedure, `findWinner`, with two parameters that are used in the procedure. The response also includes another code segment that illustrates a call to the `findWinner` procedure.
- The response describes the function of the procedure. The response states, "The function takes in the character lists and compares their average power rankings. Changing the winner screen to have an image and name of the character with the higher average but if the averages are the same a separate image and the word "`tie`" will be displayed instead." The response also explains how the procedure contributes to the overall program. The response states, "This adds to the functionality of the program as it determines the hypothetical winner in a battle between two Marvel Characters."

Create Performance Task (continued)

Row 5:
The response earned the point for this row, meeting both of the criteria:

- The student-developed algorithm within procedure, `findWinner`, includes sequencing, selection (if statement), and iteration (for loop).
- The response explains in detail how the algorithm in the procedure works so it can be recreated. The response states, "The function takes in two parameters, one list for each character selected from the drop-down box. It then declares two different variables to zero representing the sums of character rankings stored in each individual list passed through. The sum for each character is determined by iterating through the first 6 elements of the list using a for loop adding each index value to the sum variable for each character. Both of these sums are then divided by 6 to get the average ranking for each. Utilizing a conditional if-else statement the averages are compared to determine the winner. If the character selected from the first drop-down box has the higher average, the name selected from that drop-down is displayed on the winner screen along with that character's image from the URL stored in the last index of the characters list. If the character selected from the second drop-down box has a higher average, that character's name and image are displayed. If the character averages are the same the words "Tie" are displayed on the screen along with an image to represent no winner."

Row 6:
The response earned the point for this row, meeting all three criteria:

- The response describes two calls to the procedure with different arguments that lead to different results from the procedure. The first call uses the arguments Vision and Bishop. The second call uses the arguments Carnage and Venom.
- The response describes the conditions being tested for the two procedure calls. For the first call, the response states, "The first call tests the first part of the if-else statement within the `findWinner()` function, making sure that when the first character has a higher average than the character's name and the image is displayed on the winner screen. In this case, the winner screen should display the image of the character." For the second call, the response states, "The second call tests the third part of the if-else statement within the `findWinner()` function, making sure that when the averages are equal for both characters the code runs."
- The response identifies the different results of each call. For the first call, the response states, "The winner screen displays the image of Vision and the text `"Vision"` at the bottom." For the second call, the response states, "The winner screen displays the tie image and the text `"Tie"` at the bottom."

Create Performance Task (continued)

Sample Identifier: E

Score:
Row 1: 1
Row 2: 1
Row 3: 1
Row 4: 1
Row 5: 1
Row 6: 0

Row 1:
The response earned the point for this row, meeting all six criteria:

- The video demonstrates the running of the program, showing input into the program through movement of the boat as the user presses keys. Output includes movement of the boat and the score. The response states, "The video demonstrates fishing gameplay with the boat movement." This satisfies the first three criteria for the video.
- The response specifies the program's purpose is to "lessen boredom."
- The response describes the functionality demonstrated in the video. The response states, "The video demonstrates fishing gameplay with the boat movement, fishing gameplay with the hook that goes down and back to the boat to catch fish, scorekeeping with the total score calculated from fish caught in time, and scorekeeping by tracking amounts of each fish caught in time."
- The response describes the input and the output. Regarding input, the response states, "In the video, the player inputs by pressing the `'a'` and `'d'` keys." Regarding output, the response states "and the output for these keys are the boat's left and right movement."

Row 2:
The response earned the point for this row, meeting all three criteria:

- Two distinct code segments are provided. The first segment shows data being stored in the identified list, and the second segment shows data being accessed from the identified list in a for loop.
- The name of the list is identified as `fishtypes`.
- The response identifies what is stored in the list. The response states, "The data in this list represents the type of fish and the number of a specific fish caught."

Create Performance Task (continued)

Row 3:
The response earned the point for this row, meeting both criteria.

- The response includes code that uses a list of lists to manage complexity. The main list represents fish, where each list element is also a list that stores the type of fish and the number of that type of fish that are caught.
- The response explains how the code can be written without using lists in a plausible manner. The response states, “If another fish was added to the fishing game, one could simply add data of the fish into the list instead of making a new variable.” The changes needed for the program are minimal if another fish is added to the list.

Row 4:
The response earned the point for this row, meeting both of the criteria:

- The response includes a student-developed procedure, `clone+movement+range`, with five parameters that are used in the procedure. The response also includes a call to this procedure.
- The response describes the functionality of the procedure. The response states, “The procedure “`clone movement`” determines where a clone of a sprite spawns, how far it can move, and where it goes when it reaches the end of the screen.” The response also states how the procedure contributes to the overall functionality of the program by stating, “This contributes to the overall program by making it more challenging to catch a fish sprite with the hook because the procedure causes each fish clone to spawn and move randomly at a range of speeds or height.”

Row 5:
The response earned the point for this row, meeting both of the criteria:

- The student-developed algorithm within procedure, `clone+movement+range`, includes sequencing, selection (if/else statement), and iteration (for loop).
- The response explains in detail how the algorithm in the procedure works so it can be recreated. The response states, “The procedure starts by moving a newly spawned clone to a random `x` position within the screen and a random height within a range specified by the ‘`y1`’ and ‘`y2`’ parameters. Then, a loop begins until either the game ends or the clone touches the hook. Inside the loop, is an if-else statement where if the clone touches the edge of the right side of the screen, the clone would teleport left to a position specified by the ‘`rt state`’ (return state) and to random height within a range specified by ‘`y1`’ and ‘`y2`’ parameters.” Additional explanation is given.

Create Performance Task (continued)

Row 6:
The response did not earn the point for this row, meeting none of the three criteria:

- The response does not describe specific arguments that are passed through the parameters. Rather the response explains the alternate coding segments if they were used.
- The response describes the conditions being tested. The response states, “The first call tests whether the clone sprite is touching the edge of the right side of the screen.” However the response does not correspond to how the parameters are used in the program.
- The response describes the code segments rather than the result of the call.

Create Performance Task (continued)

Sample Identifier: F

Score:
Row 1: 0
Row 2: 1
Row 3: 0
Row 4: 1
Row 5: 1
Row 6: 1

Row 1:
The response did not earn the point for this row, meeting five of the six criteria:

- The video demonstrates the running of the program, showing input into the program using the keyboard, and output including the information about triangles shown on the display. This satisfies the first three criteria for the video.
- The response does not specify the program's purpose. Instead, it describes the function of the program, "to provide information about a triangle based on inputted side lengths." The response does not meet this criteria.
- The response describes the functionality demonstrated in the video. The response states, "The video demonstrates the right triangle identification functionality of the code, which includes the display of a ratio of side lengths and a table of trigonometric values."
- The response describes the input and the output. The response states, "The video demonstrates user inputs of the side lengths and program outputs of side lengths, classification by side lengths, angle measurements, classification by angle measurements, and trigonometric values of each angle."

Row 2:
The response earned the point for this row, meeting all three criteria:

- Two code segments are provided. The first segment shows data being stored in the identified list and a second segment showing data being accessed from the list.
- The name of the list is identified as `sideIndex`.
- The response identifies what is stored in the list. The response states, "The data in the list contains the possible classifications by side lengths of the triangle."

Create Performance Task (continued)

Row 3:
The response did not earn the point for this row, meeting none of the criteria:

- The response includes a list that collects the four possible classifications in one place, but this is not necessary for the program. This use of the list does not manage complexity.
- The response provides an inaccurate explanation of how the program could be written differently without lists and why lists are beneficial. The response states, "Without `sideIndex`, line 53, for example, would appear as: `return 'Classification by Side: Isosceles'`. This would be more difficult to manage than a list because in order to change an output, the coder would have to go to the exact place where that specific classification is returned, rather than being able to access all the classifications from a single list." The use of the list could be replaced with return statements with strings that produce the same effect.

Row 4:
The response earned the point for this row, meeting the two criteria:

- The response includes a student-developed procedure, `ratioCalculate`, with three parameters that are used in the procedure. The response also includes another code fragment that calls the procedure `ratioCalculate`.
- The response describes the functionality of the procedure as it states, "`ratioCalculate` takes in three inputs and returns a ratio of whole numbers which corresponds to the ratio of the inputs in the order of `A:B:C`." The procedure contributes to the overall program since it "is used to calculate the ratio of the side lengths of the triangle and is used in storing a value in `sideRatio`, which is used in the list `sideIndex`."

Row 5:
The response earned the point for this row, meeting both of the criteria:

- The student-developed algorithm within procedure, `ratioCalculate`, includes sequencing, selection (if statement), and iteration (for in range loop).
- The response explains in great detail how the algorithm in the procedure works so it can be recreated. For example, for the iterative part, the response states, "An iterative loop for variable `i` decreases by `1` from the maximum of `A`, `B`, and `C` to `0` but not including `0`. Within this loop, a remainders variable equals a concatenation by +'s of the string version of the quantity `A`." Much more detail is given for all of the other steps of the algorithm.

Create Performance Task (continued)

Row 6:
The response earned the point for this row, meeting all three criteria:

- The response describes two calls to the procedure with different arguments. The two calls are `ratioCalculate(3.0, 4.0, 5.0)` and `ratioCalculate(1.2, 2.3, 3.4)`.
- The response describes the conditions being tested for the two procedure calls. The response states for the first call that it "checks if the integer values of the inputs are equal to their actual values and can not be further simplified." The responses also states for the second call that "remainders `== "0:0:0"` is tested. This checks if `A`, `B`, and `C` can be divided by a certain value of `i` and not leave any remainders."
- The response identifies the results of each call. The response states that the first call results in `"3:4:5"`, the same values as its arguments. The response states that the second call results in `"12:23:34"` which are not the same values as its arguments.

Create Performance Task (continued)

Sample Identifier: G

Score:
Row 1: 1
Row 2: 0
Row 3: 0
Row 4: 1
Row 5: 1
Row 6: 0

Row 1:
The response earned the point for this row. The response met all of the six criteria:

- The video demonstrates the running of the program, collecting input using a text entry field and displaying output results on the screen showing letters that matched. This satisfies the first three criteria for the video.
- The response states that the program's purpose is "to test critical thinking skills."
- The response describes the functionality demonstrated in the video. The response states, "the user initially clicks the 'help' button to learn how to play. Then, it shows the user trying to guess the word by inputting an 8-letter-word into the textbox, clicking enter, then having the color scheme of the corresponding letters be outputted. The user correctly answered the first word "touching" in 4 tries so the program changed to the "You Win" screen. The user clicked play again, inputting new words but couldn't guess the correct word ("sandwich") in 6 tries."
- The response describes the input and the output. The response states, "the user inputs the word "fraction." The output displays the colors that correspond to the letters imputed."

Row 2:
The response did not earn the point for this row, meeting two of the three criteria:

- Two distinct code segments are provided. The first segment shows storage of data in a list named guesses. However, in the second code segment, data stored in the list is not being used; only the length of the list is accessed. So the response does not meet this criteria.
- The name of the list is identified as `guesses`.
- The response states, "The data contained in the list represents all of the user's word inputs."

Create Performance Task (continued)

Row 3:
The response did not earn the point for this row, meeting neither of the two criteria:

- The response includes a program code segment that uses a list, but the list is not used to manage complexity. The use of the list is merely to act as a counter, so it can be replaced with a single counter variable.
- The response does not explain how the list manages complexity. The response refers to the list `wordList`, which is not identified, so this part of the response is irrelevant. Regarding the list named `guesses`, the response states, "without the list '`guesses`,' the game would never end as the user would have infinite guesses. The point of this list is to stop the game when the user has guessed 6 times." This is implausible since the program could just use a counter.

Row 4:
The response earned the point for this row, meeting both of the two criteria:

- The response includes a student-developed procedure, `isitcorrect`, with one explicit parameter, `checkanswer`, and a call to this procedure in a second code segment using the argument, `answer`.
- The response describes the functionality of the procedure although it misnames the procedure. (Naming the procedure is not required for this row.) The response states the functionality as "checking the answer and seeing if it matches the correct word or letter positions." It also describes how it contributes to the overall program. The response states, "The algorithm is needed every time a user inputs their guess."

Row 5:
The response earned the point for this row, meeting both of the two criteria:

- The student-developed algorithm within procedure `isitcorrect` includes sequencing, selection (an if statement), and iteration (a repeat loop).
- The response explains how the algorithm works in detail. It states that "If the input is less than 9 letters, the program will not output anything. However, if the input is 8 letters, the letters will be checked for position and accuracy. If the letter is in the right position, the output letter will turn green. If the letter is in the wrong position but in the word, the output letter will turn yellow while if the letter isn't present in the word, the output letter will turn red. This part of the algorithm uses iteration as this process is repeated 8 times for the 8 letters imputed. This algorithm also determines when the 'You Win' or 'You Lose' screen will show up. … If you guessed the word or ran out of guesses, the program will switch screens."

Create Performance Task (continued)

Row 6:
The response did not earn the point for this row, meeting two of the three criteria:

- The response gives the conditions being tested rather than two different arguments that cause a different segment of code to execute. Arguments should be specific values used in the call to the procedure.
- The response describes the conditions being tested for the cases. The response states these conditions as “whether or not the user inputs the right letter in the right position” and “whether or not the user inputs the right letter in the wrong position.”
- The response describes what would result in each condition. The response states these results as “the letter color changing to green” and “the letter color changing to yellow.”

Create Performance Task (continued)

Sample Identifier: H

Score:
Row 1: 1
Row 2: 0
Row 3: 0
Row 4: 0
Row 5: 1
Row 6: 1

Row 1:
The response earned the point for this row, meeting all six criteria:

- The video demonstrates the running of the program, showing input into the program using the keyboard for the option entry and the yes/no answer, and output including the prompts and the result of the match on the display. This satisfies the first three criteria for the video.
- The response specifies the program's purpose. The response states that it can be "used to make a decision."
- The response describes the functionality demonstrated in the video. The response states, "The program prompts the user to input one of three options: Rock, Paper, or Scissors. The computer will also pick one of these three options. Depending on the relationship between the user input and the computer input, the resulting output will be one of three messages: 'You Win…', 'You Lose!', or 'It's a draw'. The user is then given the choice to play again."
- The response describes the input and the output. The response states, "The user input is either Rock, Paper, or Scissors, and the output is either 'You Win…', 'You Lose!', or 'It's a draw'."

Row 2:
The response did not earn the point for this row, meeting two of the three criteria:

- Two code segments are provided. The first segment shows data being stored in the identified list, but the second segment does not show data being accessed from the list.
- The name of the list is identified as `RPS`.
- The response identifies what is stored in the list. The response states, "The data contained in the list are the potential options that the computer can select as its input."

Create Performance Task (continued)

Row 3:
The response did not earn the point for this row, meeting neither of the two criteria:

- The response includes a list, but this list does not manage complexity in the program. This program can be written equivalently without the list using an if statement to set `cpuPlay` variable based on a random number.
- The response provides an explanation of how the program could be written differently without lists. The response states, "For the program to function without the use of list `RPS`, we would need to have each potential computer input correspond to a number. The program would then pick a random number between 1 and 4 exclusive. Depending on the number, the program would have functionally chosen one of the three potential computer inputs." But this results in a program that is just as easy to develop or maintain as the one with lists.

Row 4:
The response did not earn the point for this row, meeting one of the two criteria:

- The response includes a student-developed procedure, `rpsGame`, with one parameter that is used in the procedure. The response also includes another code fragment that calls the procedure `rpsGame`.
- The response describes the functionality of the procedure as it states, "The users input is saved as the variable `yourPlay`. This variable is used as the parameter for the function `rpsGame`. The parameter passes through the series of If, Else if statements and results in a different output depending on both the user's input and the computer's input." The response does not specify how the procedure contributes to the overall program, only stating that it "allows for the program to execute smoothly."

Row 5:
The response earned the point for this row, meeting both of the criteria:

- The student-developed algorithm within procedure, `rpsGame`, includes sequencing, selection (if statement), and iteration through use of recursion to call `rpsGame` again to execute.
- The response explains how the algorithm in the procedure works so it can be recreated. The response states, "The computer picks randomly between the items of the list, and saves the selected item in variable `cpuPlay`. The computer displays the computer's choice, and determines the output by putting the users and computers inputs through a series of If, Else if statements. The computer prints the results of the If, Else if statements. The user is offered to play again. If the user inputs `"Yes"`, the function is called again. If the user inputs `"No"`, then the computer prints `"Goodbye"`."

Create Performance Task (continued)

Row 6:
The response earned the point for this row, meeting all three criteria:

- The response describes two calls to the procedure with different arguments, leading to different results. For the first call, "`yourPlay` had the value of `"Rock"`" and "The computer input was `"Paper"`." For the second call, "`yourPlay` had the value of `"Paper"`" and "The computer input was `"Rock"`."
- The response describes the conditions being tested for the two procedure calls. In the first case, the "computer checked to see if the user's input was equal to `"Rock"` and if the computer's input was `"Paper"`." The response also states, "When it found that this statement was true," referring to the condition. In the second case, the "computer checked to see if the user's input was equal to `"Paper"` and if the computer's input was `"Rock"`. The response also states, "When it found that this statement was true," referring to the condition.
- The response identifies the results of each call. In the first case, the function displays `"You Lose!"`, and in the second case the function displays `"You Win…"`.

Create Performance Task (continued)

Sample Identifier: I

Score:
Row 1: 0
Row 2: 0
Row 3: 0
Row 4: 0
Row 5: 0
Row 6: 1

Row 1:
The response did not earn the point for this row. The response met three of the six criteria:

- The video demonstrates the running of the program, displaying icons of animals and questions, inputting the text of the guesses, and displaying the final result. This satisfies the first three criteria for the video.
- The response states that the program's purpose is "The purpose of this code is for you to see all the animals and to identify them when you're asked later" This states the function of the program and not its intended purpose. This statement does not state the problem being solved or creative interest being pursued through the program.
- The response does not describe the functionality demonstrated in the video. The response states, "Challenge the users memory to remember the animals shown on the screen." This response does not describe the functionality shown in the video.
- The response describes the input and the output; however, the response switches the input with the output and vice versa. The response states, "The input in the video is demonstrated by the user getting asked "What animals were shown on stage." The output is for the user to type what animals they saw."

Row 2:
The response did not earn the point for this row, meeting one of the three criteria:

- Two distinct code segments are provided: one shows storage of data in a list named `animalImages`; however, the second code segment shows another list named `animalList`. The response identifies `animalList`, but `animalList` is not being used in either code segment.
- The name of the list is identified as `animalList`.
- The response does describe the data contained in `animalList`, but it also describes the images from `animalImages`, stating that the images are in the list.

Create Performance Task (continued)

Row 3:
The response did not earn the point for this row, meeting none of the two criteria:

- The response does not include a program code segment that uses a list to manage complexity. Both code segments show data being stored in a list, but neither contain code using a list. The code is not complex.
- The response does not explain how the list manages complexity. The response states, "The array allows you to store more than one variable making it shorter and more complex." The response does not explain how it would be written differently without the list, and the code does not involve complexity.

Row 4:
The response did not earn the point for this row, meeting one of the two criteria:

- The response includes a student-developed procedure, `evaluateGuess`, with one explicit parameter, `guess`, and a call to this procedure in a second code segment using the argument `guess`.
- The response describes the functionality of the procedure as it states that it contributes to "determining if the user's guess was right or wrong." The response does not describe how this procedure contributes to the overall functionality.

Row 5:
The response did not earn the point for this row, meeting one of the two criteria:

- The student-developed algorithm within procedure, `evaluateGuess`, includes sequencing, and iteration (a for loop), and selection (an if statement).
- The response includes a minimal description, but it does not explain how the algorithm works in enough detail that someone else could recreate it.

Row 6:
The response earned the point for this row, meeting all of the three criteria:

- The response describes two different calls to the specific procedure to result in different code being executed. The response states, "The first call I sent was '`dog`' which was a correct guess. The response describes the second call as, "The second call I sent was '`mouse`' which was an incorrect guess."
- The response describes the conditions being tested. The response states, "The condition tested in the first call is when `evaluateGuess` is equal to one of the animals in `animalList`." The response states the other condition being tested is "when `evalulateGuess` is not equal to one of the animals in `animalList`."
- The response describes the results of the two calls, leading to two different results. The response states, "Result of the first call was 'You got a point!'," and "Result of the second call was 'Thats 1 strike'."

Create Performance Task (continued)

Sample Identifier: J

Score:
Row 1: 1
Row 2: 0
Row 3: 0
Row 4: 0
Row 5: 0
Row 6: 0

Row 1:
The response earned the point for this row. The response met six of the six criteria:

- The video demonstrates the running of the program, showing input into the program using a drop-down list of state names and displaying output results on the screen as a picture of a state flag and other data related to a state such as area and population. This satisfies the first three criteria for the video.
- The response specifies the program's purpose. The response states, "This app can help with memorization or can just be used as a way to learn something new."
- The response describes the functionality demonstrated in the video. The response states, "The video shows the user selecting the state they want to learn about and then they are directed to a screen showing general information about that state. The user also selects from a dropdown of more information and they click to view the state flag."
- The response describes the input and the output. The response states, "The input is what state the user selects and what information the user wants from the more information drop down on the general information screen." For output, the response states, "Both if these items are displayed on the general information screen."

Row 2:
The response did not earn the point for this row, meeting one of the three criteria:

- Two distinct code segments are provided, but only the second code segment shows data being stored in the identified list. No code is shown to illustrate data being used from the list.
- The name of the list is identified as `stateList`.
- The response states, "The data in the list contains all the information the user will see on the general information screen like State name, Flag, Population, Capital, and Population. This list also contains the information displayed in the "More information" drop-down." However, based on the code fragment, the description is inaccurate as the identified list `stateList` appears to represent only the state name.

Create Performance Task (continued)

Row 3:
The response did not earn the point for this row.

- The response includes code that uses lists to manage complexity; however, the response does not explain how the specific code cannot be written without a list or would be written in a more complex manner. The response states, "Without the use of a list the program code would be extraordinarily complex, all the data would have to be coded independently, taking a very long time. The code would be very inefficient without the use of a list therefore it would take much longer to run." This response is overly general and does not give specific insight into the student-written response.

Row 4:
The response did not earn the point for this row, meeting none of the two criteria:

- The response includes a student-developed procedure, `updateScreen`, but it does not have any parameter. In addition, the second code segment does not show any call to the procedure `updateScreen`.
- The response inaccurately describes the functionality of the procedure as it states, "The above procedure shows the program receiving the state input from the user and updating the state screen with the correct information for the users desired state." The procedure only sets an index and does not update the state screen. The response does not describe how this procedure contributes to the overall functionality.

Row 5:
The response did not earn the point for this row, meeting neither of the criteria:

- The student-developed algorithm within procedure `updateScreen` includes sequencing, and selection (if/else), but it does not include iteration.
- The response partially explains how the algorithm in `updateScreen` works. The response states, "Our program uses else if statements to narrows down what state the user wants to see information about." The response also states, "The algorithm then works through a series of else if statements checking to see if the state matches with the user inputs and moving on to the next it it's not a match. This continues until it reaches the state that corresponds with the user's input." The response does not state that the index value is set based on the match and what value is set for each U.S. state.

Create Performance Task (continued)

Row 6:
The response did not earn the point for this row, meeting none of the three criteria:

- The response describes two calls from the specified procedure, rather than two calls to the specified procedure. In addition, this function does not have an implicit parameter for state. Instead, this is set after the call to the procedure. An implicit parameter is one that is assigned in anticipation of a procedure call, not after the call is made.
- The response describes the two separate operations being performed by the user, not conditions being tested by the given parameter. The response states as the first condition, "The user selects their desired state from the drop down." The response states as the second condition, "The user will choose the extra information they wish to see about their state from the drop down menu."
- The response does not specify the result of two calls to the given procedure. Instead, it describes the result that appears on the screen. The response states this as "`General_Info` screen getting populated with the correct information" and "the extra information about the state the user requested from the drop down menu."