## **Written Responses (WR)**

##### **(Created Independently)**

Submit your responses to prompts 3a – 3d, which are described below. Your response to all prompts combined must not exceed 750 words (program code is not included in the word count). Collaboration is not allowed on the written responses.

* Provide written responses to the prompts in the fields indicated. You can format text using the options in the text box.
* The total word count for the written response is 750 words. To allow some flexibility, your typed responses to all prompts must not exceed 850 words. A word count is displayed for each text entry box, and an overall word count is displayed at the top and bottom of the page.
* Prompts asking for code segments must be inserted as images. The easiest way to do this is to take a screen capture (screenshot) of the appropriate code and save it. Then either drag and drop the image, or click on the photo icon and select the file. You can also copy and paste the image of the program code.
* Save your work by clicking on one of the Save buttons that are at the top and bottom of the page. Your work will be auto-saved every two minutes.

For more detailed information related to the Written Responses, please refer to the [Student Handouts](https://apcentral.collegeboard.org/pdf/ap-csp-student-task-directions.pdf?course=ap-computer-science-principles). Scoring Guidelines for the Create Performance Task can be found on the [exam page on AP Central](https://apcentral.collegeboard.org/courses/ap-computer-science-principles/exam?course=ap-computer-science-principles).

##### **Submission Reminder**

Before submitting your work as final, please:

* Do not include your name.
* Ensure that you have not collaborated on the written responses.
* Refrain from publicly posting any portion of your performance task. Work that is shared online is subject to being plagiarized by another student. When this happens both students will be flagged for plagiarism.

##### **File Upload and Submission**

#### **Alert**

Your work will **not** be sent to the College Board for scoring unless you submit your work as final.

this assignment has required sections that are empty

**SaveSubmit Final**Total words: 366

**3. WRITTEN RESPONSES (Created Independently)**

Submit your responses to prompts 3a – 3d, which are described below. Your response to all prompts combined must not exceed 750 words (program code is not included in the word count). Collaboration is **not** allowed on the written responses.

| **3 a.** | Provide a written response that does all three of the following: 164 WORDS |
| --- | --- |

*Approx. 150 words (for all subparts of 3a combined)*

| **i.** | Describes the overall purpose of the program. |
| --- | --- |

43 words

The overall purpose of the program is to teach those with ADHD, who don't know how to use their iPhone entirely, by composing lessons according to their skill levels, while incorporating a random color feature (scientifically approved method) to help with attention deficit.

| **ii.** | Describes what functionality of the program is demonstrated in the video. |
| --- | --- |

73 words

In the video, the program asks the user to add their favorite colors, through a drop-down menu. Then, only those colors will be randomly selected, every time the function, “composeLesson”, is called, and used for font-colors. This program prompts the user with a question asking about their knowledge of a certain iPhone feature. Depending on their answer choices the program will either display the lesson, or a message about how smart they are.

| **iii.** | Describes the input and output of the program demonstrated in the video. |
| --- | --- |

48 words

The inputs of the program are mouse click events on buttons for either adding colors or to move on, user-inputted text corresponding to their name, and a dropdown list selection to select their favorite colors. The outputs of my program are the randomly selected colors used for text, lesson (string) with images, and messages (string).

| **3 b.** | Capture and paste two program code segments you developed during the administration of this task that contain a list (or other collection type) being used to manage complexity in your program. |
| --- | --- |

*Approx. 200 words (for all subparts of 3b combined, exclusive of program code)*

| **i.** | The first program code segment must show how data have been stored in the list. 202 WORDS |
| --- | --- |

**Drop Images Here**

**click to delete image**

Thumbnail of an image whose file name is, AddColor2Capture.PNG

| **ii.** | The second program code segment must show the data in the same list being used, such as creating new data from the existing data or accessing multiple elements in the list, as part of fulfilling the program's purpose. |
| --- | --- |

**Drop Images Here**

**click to delete image**

Thumbnail of an image whose file name is, listUsecapture2.PNG

Then, provide a written response that does all three of the following:

| **iii.** | Identifies the name of the list being used in this response. |
| --- | --- |

7 words

The name of the list is "colorArray".

| **iv.** | Describes what the data contained in the list represent in your program. |
| --- | --- |

28 words

The data in, "colorArray", represents the colors selected by the user. Colors are randomly selected from this list to be used as font-colors for various messages and lessons.

| **v.** | Explains how the selected list manages complexity in your program code by explaining why your program code could not be written, or how it would be written differently, if you did not use the list. |
| --- | --- |

167 words

The list, "colorArray", manages complexity in this program by allowing me to refer to the entire list of colors, using a single variable name. This allows me to generate one random color using just one line of code, that selects a color from the list based on a randomly generated index (as seen in "function randomColor()"). If we didn't have this list, each of the elements in the list would need its own variable and name. Generating a random color from numerous variables would require me to write my code in a less efficient manner. This is because my code would have to refer to each of these 7 variables individually rather than being able to reference elements by their index in the list. Also, if I later decide I want to add an 8th color to "colorArray", I would have to completely rewrite my code. However, by using a list, all I have to do is modify the line of code where the color can be added to the list.

| **3 c.** | Capture and paste two program code segments you developed during the administration of this task that contain a student-developed procedure that implements an algorithm used in your program and a call to that procedure. |
| --- | --- |

*Approx. 200 words (for all subparts of 3c combined, exclusive of program code)*

| **i.** | The first program code segment must be a student-developed procedure that: |
| --- | --- |

* Defines the procedure's name and return type (if necessary)
* Contains and uses one or more parameters that have an effect on the functionality of the procedure
* Implements an algorithm that includes sequencing, selection and iteration

**Drop Images Here**

**click to delete image**

Thumbnail of an image whose file name is, procedureCode2.PNG

| **ii.** | The second program code segment must show where your student-developed procedure is being called in your program. |
| --- | --- |

**Drop Images Here**

Then, provide a written response that does both of the following:

| **iii.** | Describes in general what the identified procedure does and how it contributes to the overall functionality of the program. TOTAL IS 248 |
| --- | --- |

0 words

The, “composeLesson()”, function takes the skill level of the user (Boolean value) and lesson number as input. “composeLesson()” contributes to the overall functionality of the program by returning a personalized lesson or message that includes the random colors used for text, to help with attention-deficit, and uses a for-loop to make sure colors don’t appear next to each other. If the user knows the information in a lesson, which will be identified with the questions asked in the program, then the program will display a message about how smart they are. If the user doesn’t know the information in the lesson, then it displays them the lesson.

| **iv.** | Explains in detailed steps how the algorithm implemented in the identified procedure works. Your explanation must be detailed enough for someone else to recreate it. |
| --- | --- |

0 words

The, “composeLesson()”, function uses a for-loop that assigns a variable a random color and splices that color from, “colorArray”, during each iteration. Then, “composeLesson “ uses an if-else statement to decide which lesson to display, and which message in the lesson to display. “composeLesson()” uses the argument passed on the parameter, “lessonNumber”, to decide which lesson to display. “composeLesson()”, then uses the argument passed on to the parameter, “skill”,(Boolean value) in another if-else statement. If “skill” is equal to true, then the function concatenates a string, the random color variable (Which is in a span tag), and a gif, which will become the message that tells the user how smart they are, else, the function concatenates a string, the random color variable, and an image which will become the personalized lesson. Finally, the function returns the string.

| **3 d.** | Provides a written response that does all three of the following: |
| --- | --- |

*Approx. 200 words (for all subparts of 3d combined)*

| **i.** | Describes two calls to the 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. 206 WORDS |
| --- | --- |

First call:

0 words

Our first call is, “composeLesson(‘lessonOne’, skillCheck)”. We pass 2 arguments: ‘lessonOne’, representing the lesson number, and skillCheck, which is a boolean value.

Second call:

0 words

Our first call is, “composeLesson(‘lessonTwo’, skillCheck)”. We pass 2 arguments: ‘lessonTwo’, representing the lesson number, and skillCheck, which is a boolean value.

| **ii.** | Describes what condition(s) is being tested by each call to the procedure. |
| --- | --- |

Condition(s) tested by first call:

0 words

“composeLesson(‘lessonOne’, skillCheck)” tests which lesson the user is on, in the case of this call it’s ‘lessonOne’. Another condition being tested is the value of “skillCheck” (true or false), which is dependent on the user's response to a question about their skill.

Condition(s) tested by second call:

0 words

“composeLesson(‘lessonTwo’, skillCheck)” tests which lesson the user is on, in the case of this call it’s ‘lessonTwo’. Another condition being tested is the value of “skillCheck” (true or false), which is dependent on the user's response to a question about their skill.

| **iii.** | Identifies the result of each call. |
| --- | --- |

Result of the first call:

0 words

Since the first call’s first argument asks for ‘lessonOne’, “composeLesson(‘lessonOne’, skillCheck)”, displays either lesson one or a message from lesson one depending on the value of “skillCheck”, which comes from the user's response to a question about their skill.

Result of the second call:

Since the second call’s first argument asks for ‘lessonTwo’, “composeLesson(‘lessonTwo’, skillCheck)”, displays either lesson two or a message from lesson two depending on the value of “skillCheck”, which comes from the user's response to a question about their skill.