Video
Submit one video in .mp4, .wmv, .avi, or .mov format that demonstrates the running of at least one significant feature of your program. Your video must not exceed 1 minute in length and must not exceed 30MB in size.

Program Purpose and Development
2a. Provide a written response or audio narration in your video that:
   - identifies the programming language
   - identifies the purpose of your program; and
   - Explains what the video illustrates.
   (Must not exceed 150 words)

Student Response
https://youtu.be/uKVGOZv4H1g
My program is an interactive game created with Javascript. The purpose of this game is to create something fun that interacts with the user by incorporating an adventure that the player completes by performing certain tasks or winning certain games

<table>
<thead>
<tr>
<th>Total score</th>
<th>Row 1</th>
<th>Row 2</th>
<th>Row 3</th>
<th>Row 4</th>
<th>Row 5</th>
<th>Row 6</th>
<th>Row 7</th>
<th>Row 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample: C</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

This document combines student sample, scoring guidelines and scoring commentary from: Create PT Sample C

### Scoring Guidelines

<table>
<thead>
<tr>
<th>Row and Task</th>
<th>Decision Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Row 1</strong> Response 2A</td>
<td>The video demonstrates the running of at least one feature of the program submitted. AND The response (audio narration or written response) identifies the purpose of the program (what the program is attempting to do).</td>
</tr>
<tr>
<td>Response earns the point if it explains the function of the program instead of identifying the purpose.</td>
<td></td>
</tr>
<tr>
<td>Response earns the point if the illustrated feature runs, even if it does not function as intended.</td>
<td></td>
</tr>
<tr>
<td>Response earns the point if the video includes a narration or some form of closed captioning that addresses the purpose of the program.</td>
<td></td>
</tr>
</tbody>
</table>

Do NOT award a point if any one of the following is true:
- a video is not submitted;
- the video does not illustrate the feature mentioned in the response; or
- the video does not illustrate the running of the feature (screen shots or storyboards are not acceptable and would not be
within each adventure. Once all of the subgames on each level of the app are won, the user completes the adventure. The video shows one of two adventures that are incorporated in my game to display how the program tracks score within each mini game and on each overall level/adventure to determine when the overall game/adventure is won. The video also shows the functionality of the games and how they are played, in addition to the storyline format that the adventures follow. (124 words)

The response earned a point for this row.
The video is continuously running and demonstrates two adventures in the adventure game. The response indicates that the purpose of this program is to create a fun and interactive adventure game.

2b. Describe the incremental and iterative development process of your program, focusing on two distinct points in that process. Describe the difficulties and/ or opportunities you encountered and how they were resolved or incorporated. In your description clearly indicate whether the development described was collaborative or independent. At least one of these points must refer to independent program development. (Must not exceed 200 words)

<table>
<thead>
<tr>
<th>Student Response</th>
<th>Scoring Guidelines</th>
</tr>
</thead>
</table>
| I wrote this program, first by creating an identity for the user by incorporating user choice and a user input box to allow for personalization, and then by creating worlds with mini games within them for the user's chosen character to complete. My first problem arose when I tried to carry the username input from the character choice screen throughout the rest of the game. Because I had initially used a local variable, the user input was only recorded to the screen on which it was provided, instead of to the entire program. To resolve this issue, I changed the username input to a global variable that was able to be called on multiple screens in the program. I encountered a similar issue when I attempted to move the user's chosen character between screens. To make it appear as though the image traveled to the new screen as the user switched screens, I created a variable that stored the value of whichever character the user clicked on the first screen (i.e. boy= 1 when the boy image was clicked) and then
| Row 2 - Response 2B | Do NOT award a point if any one of the following is true: |
| Describes or outlines steps used in the incremental and iterative development process to create the entire program. | ● the response does not indicate iterative development;  |
| | ● refinement and revision are not connected to feedback, testing, or reflection; or |
| | ● the response only describes the development at two specific points in time. |
| The response earned a point for this row. | Response earns the point if it identifies two opportunities, or two difficulties, or one opportunity and one difficulty AND describes how each is resolved or incorporated. |
| The response describes the development process to create the entire program. The response states, "I wrote this program, first by creating an identity for the user… and then by creating worlds with mini games within them for the user's chosen character to complete." The response describes how they used an iterative development process by testing the use of the username throughout the game. "To resolve this issue, I changed the username input to a global variable that was able to be called on multiple screens in the program." | Do NOT award a point if any one of the following is true: |
used an algorithm that incorporated an if statement to set other images throughout the program to the user's character of choice. Both of these difficulties were handled independently. (210 words)

<table>
<thead>
<tr>
<th>AND</th>
<th>Describes how the two identified difficulties or opportunities are resolved or incorporated.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• only one distinct difficulty or opportunity in the process is identified and described; or</td>
</tr>
<tr>
<td></td>
<td>• the response does not describe how the difficulties or opportunities were resolved or incorporated.</td>
</tr>
</tbody>
</table>

The response earned a point for this row. The response describes two difficulties that were encountered and how these were resolved. The first difficulty is with the carrying of inputs to a different screen. This is resolved by using a global variable instead of a local variable. The second difficulty is the moving of the user's character between screens. This is resolved by using a variable and an if statement to set the image of the user's character throughout the program.

2c. Capture and paste a program code segment that implements an algorithm (marked with an oval in section 3 below) and that is fundamental for your program to achieve its intended purpose. This code segment must be an algorithm you developed individually on your own, must include two or more algorithms, and must integrate mathematical and/or logical concepts. Describe how each algorithm within your selected algorithm functions independently, as well as in combination with others, to form a new algorithm that helps to achieve the intended purpose of the program. (Must not exceed 200 words)

```javascript
1. A.
   onEvent("coin", "click", function(event)
      treasureScore = treasureScore + 1;
      randomNumber(105, 220);
      setPositon("coin", randomNumber(190, 250));
      setText("coinCollected", "Number of coins collected:
      +treasureScore");
      if (treasureScore == 20)
         setScreen("treasureWin");
         getCoins = 1;
         });
   onEvent("treasureContinue", "click", function(event) {
      treasureScore = 0;
      setScreen("underwaterAdventures");
      didWinUnderwater();
      if (getCoins==1) {
         setText("coins", "Coins? Yes");
      });
   });

2. function didWinUnderwater()
   { if (getCoins==1&&scale==1) {
      setPositon("fishTalk", 25, 150, 210, 130);
      showElement("fishTalk");
      setText("fishTalk", "You have collected all of the elements you need to deliver to the shark. Would you like
      to proceed?";)
      showElement("yesUA");
      showElement("noUA");
      setPositon("underwaterCharacter", 250, 205);
      reset(getCoins, scale);
      }
```
In the adventure "Underwater Adventures" there is a mini game in which the user must collect 20 coins to win. Once this task is complete, the user must play a second mini game in which he/she collects a scale. When both mini games are won, the Underwater Adventures level is complete. The code to execute this concept incorporates a variety of variables and functions to create an algorithm that calculates when the treasure game and scale games have been won, and when the overall level has been completed. The first set of code shown above (1.A.) is the function that runs the treasure game. Each time the coin is clicked, the variable treasureScore increases by 1. Once the score reaches 20, another variable, gotCoins, increases by 1 and the screen switches to a win screen with a continue button. The scale game works similarly, with a variable scale increasing by one if the game is won (1.B.). If the user clicks the continue button after winning either game, the second piece of code shown (2.) uses an algorithm to check if both games have been won by calling the function didWinUnderwater, which uses an if statement to determine if both gotCoins and scale have the value of 1. (211 words)

<table>
<thead>
<tr>
<th>Student Response</th>
<th>Scoring Guidelines</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Row and Task</strong></td>
<td><strong>Decision Rules</strong></td>
</tr>
</tbody>
</table>
| In the adventure “Underwater Adventures” there is a mini game in which the user must collect 20 coins to win. Once this task is complete, the user must play a second mini game in which he/she collects a scale. When both mini games are won, the Underwater Adventures level is complete. The code to execute this concept incorporates a variety of variables and functions to create an algorithm that calculates when the treasure game and scale games have been won, and when the overall level has been completed. The first set of code shown above (1.A.) is the function that runs the treasure game. Each time the coin is clicked, the variable treasureScore increases by 1. Once the score reaches 20, another variable, gotCoins, increases by 1 and the screen switches to a win screen with a continue button. The scale game works similarly, with a variable scale increasing by one if the game is won (1.B.). If the user clicks the continue button after winning either game, the second piece of code shown (2.) uses an algorithm to check if both games have been won by calling the function didWinUnderwater, which uses an if statement to determine if both gotCoins and scale have the value of 1. (211 words) | Do NOT award a point if any one of the following is true:  
● the algorithm consists of a single instruction;  
● the code segment consisting of the algorithm is not included in the written responses section or is not explicitly identified in the program code section; or  
● 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 response earned a point for this row.  
The selected code segment implements an algorithm.

| Row 4  
Response 2C  
Selected code segment implements an algorithm. | The algorithm being described can utilize existing language functionality, or library calls. Response earns the point even if the algorithm was not newly developed. (i.e., a student’s reimplementation of the algorithm to find the minimum value) |
|---------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|
| Do NOT award a point if any one of the following is true:  
● the selected algorithm consists of a single instruction;  
● the selected algorithm consists solely of library calls to existing language functionality;  
● the selected algorithm does not include mathematical or logical concepts;  
● the response only describes what the selected algorithm does without explaining how it does it;  
● the response does not explicitly address the program’s purpose;  
● the code segment consisting of the selected algorithm is not included in the written responses section or is not explicitly identified in the program code section; or  
● 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 selected algorithms use logical concepts by including if statements. The response explains how the coin event algorithms functions. The response states, “Each time the coin is clicked, the variable treasureScore increase by 1. Once the score reaches 20, another variable gotCoins, increase by 1 and the screen switches
to a win screen with a continue button. The response describes how this algorithm does in relation to the overall purpose. It states, “Once this task (the coin event game) is complete, the user must play a second mini game in which he/she collects a scale. When both mini games are won, the Underwater Adventure level is complete.”

<table>
<thead>
<tr>
<th>Row 6</th>
<th>Response 2C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Selected code segment implements an algorithm that includes at least two or more algorithms. AND At least one of the included algorithms uses mathematical or logical concepts. AND Explains how one of the included algorithms functions independently.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Do NOT award a point if any one of the following is true:</th>
</tr>
</thead>
<tbody>
<tr>
<td>● the selected algorithm consists of a single instruction;</td>
</tr>
<tr>
<td>● the selected algorithm consists solely of library calls to existing language functionality;</td>
</tr>
<tr>
<td>● neither of the included algorithms nor the selected algorithm that includes two or more algorithms uses mathematical or logical concepts;</td>
</tr>
<tr>
<td>● the code segment consisting of the algorithm is not included in the written responses section or is not explicitly identified in the program code section; or</td>
</tr>
<tr>
<td>● 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).</td>
</tr>
</tbody>
</table>

The response DID NOT earn a point for this row. While three algorithms are shown, there is not an algorithm that is using at least two or more algorithms. The two algorithms independently call on the third.

2d. Capture and paste a program code segment that contains an abstraction you developed individually on your own (marked with a rectangle in section 3 below). This abstraction must integrate mathematical and logical concepts. Explain how your abstraction helped manage the complexity of your program. *(Must not exceed 200 words)*
Student Response

My code uses abstraction by incorporating a variety of functions to condense repeated code into smaller pieces. These functions can then be called within other functions or onEvents to make the code easier to read and understand. The function setUpBlastOff (1.) calls the functions hideText (2.), showText (3.), and setCharacter (4.), which are used to reduce the number of

1. ```javascript
function setUpBlastOff() {
    mars = 0;
    jupiter = 0;
    hideText(“alienTalk”, “okMars”);
    hideText(“alienTalk”, “okJupiter”);
    setText(“saveMars”, “Saved Mars? No”);
    setText(“saveJupiter”, “Saved Jupiter? No”);
    setPosition(“spaceCharacter”, 90, 285);
    setCharacter(“spaceCharacter”);
    showText(“alienTalk”, “playBlastOff”);
    showElement(“blastOffInstructions”);
    setText(“alienTalk”, (“Hello” + “username”) + “! All of the planets in space are becoming uninhabitable! I guess I’ll just have to invade Earth.”);
    setText(“blastOffInstructions”, “Prevent the alien from invading Earth by making the other planets in the galaxy safe for life. Click on the planets to explore them.”);
}
```

2. ```javascript
function hideText(textArea, button) {
    hideElement(textArea);
    hideElement(button);
}
```

3. ```javascript
function showText(textArea, button) {
    showElement(textArea);
    showElement(button);
}
```

4. ```javascript
function setCharacter(image) {
    if (boyClick >= 1) {
        setImageURL(image, “https://www.carlstickers.com/prodimages/1033_blue_sky_family_boy_stick_figure_sticker_decal.gif”);
    } else {
        setImageURL(image, “https://www.carlstickers.com/prodimages/1032_blue_sky_family_girl_stick_figure_sticker_decal.gif”);
    }
}
```

Scoring Guidelines

Row 7

Response 2D

Selected code segment is a student-developed abstraction.

Row and Task | Decision Rules
--- | ---

Responses that use existing abstractions to create a new abstraction, such as creating a list to represent a collection (e.g., a classroom, an inventory), would earn this point.

Do NOT award a point if any one of the following is true:

- the response is an existing abstraction such as variables, existing control structures, event handlers, APIs;
- the code segment consisting of the abstraction is not included in the
lines of code contained within the function. The setCharacter function (4.) is also an abstraction because it uses an if statement to check the value of the two character variables (boy and girl) and uses this logic to set the on-screen character to the one that the user chose. Overall, the function setUpBlastOff is an abstraction because it is called multiple times in the program to reset the 321BlastOff screen when the level is either won or quit. This prevents the programmer from having to retype the same several lines of code each time they want to reset the screen. These abstractions make my program more manageable because they take repeated sections of code that would add significant complexity to my algorithms and reduce them into single functions. For example, the setUpBlastOff function would require at least 12 lines of code each time it was called if it were not condensed into a function. (213 words)

<table>
<thead>
<tr>
<th>The response earned a point for this row.</th>
</tr>
</thead>
<tbody>
<tr>
<td>The selected code segment includes a student-developed procedure setUpBlastOff. The additional procedures that are shown are used in this procedure.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Row 8 Response 2D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explains how the selected abstraction manages the complexity of the program.</td>
</tr>
</tbody>
</table>

Responses should not be penalized for explanations of abstractions that are not developed by the student.

Do NOT award a point if any one of the following is true:
- the explanation does not apply to the selected abstraction; or
- the abstraction is not explicitly identified (i.e., the entire program is selected as an abstraction, without explicitly identifying the code segment containing the abstraction).

The abstraction being described is a procedure setUpBlastOff that calls three additional procedures. The response explains that the abstraction makes the program more manageable “because they take repeated sections of code that would add significant complexity to my algorithms and reduce them into single functions. For example, the setUpBlastOff function would require at least 12 lines of code each time it was called if it were not condensed into a function.”

3. Program Code
Capture and paste your entire program code in this section.

- Mark with an oval the segment of program code that implements the algorithm you created for your program that integrates other algorithms and integrates mathematical and/or logical concepts.
- Mark with a rectangle the segment of program code that represents an abstraction you developed.
- Include comments or acknowledgments for program code that has been written by someone else.