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

[Video still image]
https://youtu.be/zT1XNAS4oGE

This program was created using JavaScript. It's intended to be a turn-based game where players can progress through levels by gaining EXP. The video I've provided displays one of the main and essential features of my program, the attack system. At the start, I showed you my starting health, EXP, and gold. Then, I went into level two and displayed the stat check button and both attack buttons. The celestial attack
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>
| **This entire program was undertaken independently. I ran into several complications among the code that I had to figure out on my own. One of the issues that I came across involved configuring how I was going to unlock levels appropriately whenever a player gets a certain amount of EXP. At first, I was going to make a set of code that would keep checking if the amount of EXP reached a required amount or not. I found that it was more efficient just to create a function and run it whenever it was necessary. Now, when a player returns back to the level screens, the program will check to see if the EXP has reached the highest level, and if not, it checks the next highest level and so on. I also had to figure out how I was going to adjust the enemy's health correctly for each level. I decided to make a variable that would be the enemy's health for all levels and just set the value whenever a level was entered. On top of that, a function was...** | **Row and Task**

**Row 2 - Response 2B**

Describes or outlines steps used in the incremental and iterative development process to create the entire program.

**Decision Rules**

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

- 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 **DID NOT earn a point for this row.**

The response does not describe or outline the steps used in the development of the entire program.

**Row 3 - Response 2B**

Specifically identifies at least two program development difficulties or opportunities.

**AND**

Describes how the two identified difficulties or opportunities are resolved or incorporated.

**Decision Rules**

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.

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

- only one distinct difficulty or opportunity in the process is identified and described; or
- the response does not describe how the difficulties or opportunities were resolved or incorporated.

The response **earned a point for this row.**

The response describes two difficulties are encountered and how both were resolved. The first difficulty...
made to update the display of the new level and show the appropriate values for everything on screen. (200 words)

described is how to unlock levels when a certain amount of EXP is reached. This is resolved by creating a function that checks to see if the EXP has reached the highest level. The second difficulty described is the tracking of the health of the enemy. This is resolved by using a variable and a function to update the variable accordingly.

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
onEvent("twiAtk6", "click", function(event) {
    playerAttack(85, 65, 90, 100, "eneHPVal6", "youMiss6");
    if (health <= 0) {
        playerDeath(100);
    } else if (enehealth <= 0) {
        enemyDeath(5000, 1500, 10000);
    } else {
        enemAttack(91, 200, 93, 260, "healthVal6", "hpLose6", "atkMsg6", "missMsg6");
    }
});
```

Student Response

<table>
<thead>
<tr>
<th>Row and Task</th>
<th>Scoring Guidelines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row 4</td>
<td>Do NOT award a point if any one of the following is true:</td>
</tr>
<tr>
<td>Response 2C</td>
<td>- the algorithm consists of a single instruction;</td>
</tr>
<tr>
<td></td>
<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></td>
<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 earned a point for this row.

The selected code segment for twiAtk6 implements an algorithm.

Row 5

Response 2C

The algorithm being described can utilize existing language functionality, or library calls. Response earns the point even if the algorithm was not
kind of damage is added on from any weapons you might have. Then, it
checks to see if the enemy’s health is at 0 yet and if it is, it will send you to the
level screen again and reward you with gold, EXP, and health. If your health is at
0, it will only give you some health. Otherwise, the function enemyAttack is
run and the enemy rolls a number and attacks you instead, also with a chance
of a critical and a miss. (199 words)

<table>
<thead>
<tr>
<th>Row 6</th>
<th>Response 2C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected code segment implements an algorithm that uses mathematical or logical concepts. <strong>AND</strong> Explains how the selected algorithm functions. <strong>AND</strong> Describes what the selected algorithm does in relation to the overall purpose of the program.</td>
<td>newly developed. (i.e., a student’s reimplementation of the algorithm to find the minimum value)</td>
</tr>
</tbody>
</table>

**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 response earned a point for this row.
The selected algorithm includes logical concepts through the use of an if statement.
The response explains how the algorithm functions. The response states, “The code starts out by initiating playerAttack… Then, it checks to see if the enemy’s health is at 0 yet, and if it is, it will send you to the level screen again and reward you with gold, EXP, and health. If your health is at 0, it will only give you some health. Otherwise, the function enemyAttack is run…”
The response describes what the algorithm does in relation to the overall program. The response states, the algorithm is used “in order to fulfill working attack functions for all levels in the game.”

<table>
<thead>
<tr>
<th>Row 6</th>
<th>Response 2C</th>
</tr>
</thead>
</table>
| 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 | 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;  
- neither of the included algorithms nor the selected algorithm that includes two or more algorithms uses mathematical or logical concepts;  
- 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). |
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)

```javascript
function levelUnlock() {
    if (exp >= 10000) {
        hideElement("sixLk");
        showElement("sixOpen");
    } else if (exp >= 8000) {
        hideElement("fiveLk");
        showElement("fiveOpen");
    } else if (exp >= 5000) {
        hideElement("fourLk");
        showElement("fourOpen");
    } else if (exp >= 1500) {
        hideElement("threeLk");
        showElement("threeOpen");
    } else if (exp >= 500) {
        hideElement("twoLk");
        showElement("twoOpen")
    }
}
```

**Student Response**

An abstraction is something that can be used to compress code down and simplify it. This means we do not have to repeat a long

<table>
<thead>
<tr>
<th>Student Response</th>
<th>Scoring Guidelines</th>
</tr>
</thead>
<tbody>
<tr>
<td>An abstraction is something that can be used to compress code down and simplify it. This means we do not have to repeat a long</td>
<td><strong>Row and Task</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Row 7</strong></td>
</tr>
</tbody>
</table>
block of code over and over again and repeat it in the program. Instead, we can create an abstraction and simplify the process a lot more. One of the abstractions in my program is the levelUnlock() function. Every time the program returns to the level selection screen after completing a level, it runs this abstraction and goes through the requirements for each level to be unlocked. Now, instead of having to individually go through and check the EXP requirements for every level whenever a level is completed, I can easily use the abstraction levelUnlock() to simplify the complexity of my program. Before I came up with this abstraction to use in my program, I was going to have to create a big else if statement for every time the enemy was attacked and their health reached zero which would send them back to the selection screen. The else if statement would have been redundant to keep repeating for every attack button in the program, so crafting an abstraction made it a lot easier to manage. (200 words)

<table>
<thead>
<tr>
<th>Response 2D</th>
<th>as creating a list to represent a collection (e.g., a classroom, an inventory), would earn this point.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected code segment is a student-developed abstraction.</td>
<td>Do NOT award a point if any one of the following is true:</td>
</tr>
<tr>
<td></td>
<td>● the response is an existing abstraction such as variables, existing control structures, event handlers, APIs;</td>
</tr>
<tr>
<td></td>
<td>● the code segment consisting of the abstraction is not included in the written responses section or is not explicitly identified in the program code section; or</td>
</tr>
<tr>
<td></td>
<td>● 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).</td>
</tr>
</tbody>
</table>

The response earned a point for this row. The selected code segment is a student-developed function levelUnlock.

<table>
<thead>
<tr>
<th>Row 8 Response 2D</th>
<th>Responses should not be penalized for explanations of abstractions that are not developed by the student.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explains how the selected abstraction manages the complexity of the program.</td>
<td>Do NOT award a point if any one of the following is true:</td>
</tr>
<tr>
<td></td>
<td>● the explanation does not apply to the selected abstraction; or</td>
</tr>
<tr>
<td></td>
<td>● 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).</td>
</tr>
</tbody>
</table>

The response earned a point for this row. The response explains how the abstraction manages complexity. The response states, “Instead of having to individually go through and check the EXP requirements for every level whenever the level is completed, I can easily use the abstraction levelUnlock() to simplify the complexity of my program.”