Create – Applications From Ideas
Written Response Submission Template

Please see Assessment Overview and Performance Task Directions for Student for the task directions and recommended word counts.

Program Purpose and Development

2a)

For this game, I used HTML and CSS to build and style the website and Javascript to make the actual game. This is a simple text based choose your own adventure game in which the player follows a plot line whilst answering riddles in an effort to get to a happy ending. The video displays a quick playthrough of the game. The user starts by entering their name into the game. Then, they can read the storyline and answer riddles to continue on. As long as their answer contains a key word without which their answer would be incorrect, they move on. If they answer incorrectly, they must restart the adventure. The user can also restart the game at any point. All of this happens without moving through different HTML pages.

2b)

While making this game, I faced various challenges. A big challenge I faced was simply coming up with a story for my game. I had little idea of what I wanted to do. However, I've started to draw inspiration from different books, movies, TV shows, and video games that I've come across in order to weave an original story. It was difficult to keep track of all the different branches of my story since the player has to be able to make different decisions based on each scenario. In order to keep track of all these different branches, I wrote my story on a digital whiteboard, which I used as a way to organize all the different scenarios and options like a tree. There were also some dilemmas I faced in programming the game. For instance, variable scope was something that got in the way very often. At the very start of the game, the player is asked to enter a name. It took a lot of deleting, rewriting, and some mild research to get the 'name' variable to hold the string value that the user inputed when the game moved onto the next page.

2c)

```
function onwards(option){
  if (option == welcome){
    game.innerHTML = welcome;
    document.body.style.backgroundColor = "white";
    document.body.style.color = "black";
    ch = 0; pt = 0;
  } else if (pos(0, 0)){
    ch++;
    game.innerHTML = contents[ch][pt] + restart;
  } else if(ch != 0 && pt == 0){
    option == 1 ? pt = 1 : pt = 2;
    game.innerHTML = contents[ch][pt] + restart;
  } else if (pt == 1){
    ch++; pt = 0;
    game.innerHTML = contents[ch][pt] + restart;
  }
  game.style.paddingTop = marginTop + "px";
  listeners();
}

function pos(x, y) {
  if (ch == x && pt == y) return true;
  else return false;
}

function check() {
  var ans;
  answer = String(document.getElementById('answer').value);
  // Answer cases ////////////
  if (pos(1, 0)) ans = "cards";
  if (pos(2, 0)) ans = "promise";
  if (pos(3, 0)) ans = "joke";
  if (pos(4, 0)) ans = "river";
  if (pos(5, 0)) ans = "key";
  ////////////////////////////
  var checkAnswer = answer.includes(ans);
  if (checkAnswer) onwards(1);
  else onwards(2);
}
```

I created three main interrelated functions in order for the player to progress through the game. The function pos() checks the values of variables ‘ch’ and ‘pt’ that track the location of the user throughout the game. Based on the parameters entered, it will return true if the user is at that location or false if the user is not at that location. Based on the different locations that the user is at, the function onwards(), when called by a button that the user must click, simply moves the user along as needed. It does so by increasing the values of variables ‘ch’ and ‘pt,’ which is where it uses math. Finally, the function check() is used to check the user’s answer to a riddle or question. There is a list of the correct answers that correspond with which location in the game. If the user’s answer includes the correct answer as a substring anywhere in their response, they are allowed to move on. If not, they are taken to a game over page and made to restart once again.

2d)

```
function onwards(option){
  if (option == welcome){
    game.innerHTML = welcome;
    document.body.style.backgroundColor = "white";
    document.body.style.color = "black";
    ch = 0; pt = 0;
  } else if (pos(0, 0)){
    ch++;
    game.innerHTML = contents[ch][pt] + restart;
  } else if(ch != 0 && pt == 0){
    option == 1 ? pt = 1 : pt = 2;
    game.innerHTML = contents[ch][pt] + restart;
  } else if (pt == 1){
    ch++; pt = 0;
    game.innerHTML = contents[ch][pt] + restart;
  }
  game.style.paddingTop = marginTop + "px";
  listeners();
}
```

The onwards() function is really important in managing my game as it is what allows the user to move forward as they need to. This function abstracted away a lot of what I'd need to worry about, such as tracking where the user was in the game and where they should move forwards based on their location and, if applicable, their game responses. It continuously updates the 'ch' and 'pt' variables that are important to other functions as well, which means that no matter what, my program will always be able to locate the user by using those two variables.