**Criterion C: Development**  (30 points) The product must be compatible with the information in criterion A and criterion B.

Present a list of the techniques used in developing the product.  The techniques may include algorithmic thinking, data structures, software tools and user interface. This list need not be exhaustive but should illustrate how the major components of the product were developed.

Software Tools:

* JavaScript
  + The programming language used to create the game scene
* HTML
  + Functions to display new background images as playable areas
  + Creating buttons on canvas to navigate between images
  + Defines canvas size
* Visual Studio Code
  + Used Live Share to collaborate on a shared document of code
* Google Chrome
  + Area where HTML files were placed

Data Structure and Algorithmic Thinking:

* Five button interface
  + Start
    - Using an onclick command, when the mouse button is clicked on this start button, it calls a function that changes the background image to the game layout, as well as beginning the actual game function itself
  + Controls
    - Using an onclick command, when the mouse button is clicked on this control button, it calls a function in which it changes the canvas image to the control screen image that details the controls needed to play the game
  + How to Play
    - Using an onclick command, when the mouse button is clicked on this How to Play button, it calls a function in which it changes the canvas image to the image that details the rules behind the game Joust and how to play it
  + Highscore
    - Using an onclick command, when the mouse button is clicked on this Highscore button, it calls a function in which it changes the canvas image to an image that contains the fastest time a game has been completed in by comparing the time of each completed time to the high score time and updating the High Score if the completed time is lower
  + Quit
    - Using an onclick command, when the mouse button is clicked on this quit button, it calls a function in which it changes the canvas image to nothing, thus quitting out of the game
* Scoreboard
  + The scoreboard is an image that is shown after each round is completed, in which the values are based upon what the game ended with. For example, there are respective areas for each player that details the amount of lives each player ended the game with, as well as a time box that displays the time the game was completed in
* Controlling the Character:
  + Function watches for key press and then updates the sprite depending on the key:
  + “W” key or “Up” arrow updates the sprite’s x and y speed based on the direction the sprite is facing and changes the sprite to flapped.
  + “A” key or “Left” arrow updates the sprite’s image and direction facing to the left
  + “D” key or “Right” arrow updates the sprite’s image and direction facing to the right
  + The animate function changes the sprite’s x and y according to the x and y speed of each
* Gravity
  + Each time through the animate function, the y speed is decreased by 0.25 and therefore slowly pushes the sprites back down the screen
* Collision and Hitboxes
  + Each sprite has a rectangle which mirrors the sprite’s x and y, by checking if the two hitboxes are colliding, we can know if the two players are touching.
  + Once touching, we check which y axis of the player is higher, and the lower player loses a life.
* Sound
  + A sound is played each time a player loses a life, this is done by playing an audio file from the folder in which the file is located
* Life Counter
  + Displayed below each player’s spawn point is a life counter of three
  + Everytime the player loses a life, both players’ x and y are set to the spawn and the life counter of the losing individual is decreased by one.
* Boundaries
  + Used the code behind the game “Pong” as a reference in order to create a working collision factor between players
    - Also used the boundaries of Pong, however changed the boundaries to teleport the player to the opposite side instead of reversing the direction of the player
* Bottom and Ceiling Collision
  + As the players hit the bottom they softly bounce off, this is done by changing the y speed by 1.
  + As the players hit the top they strongly bounce off forcing players to stay near the bottom, this is done by changing y speed by 5.
* Theorycrafting
  + Used fundamental rules of Joust as a foundation to expand upon this project

A testing plan that addresses the main areas of functionality of the product.  The test plan should address both Clear Box and Black Box testing.

* Black Box Testing

|  |  |
| --- | --- |
| Tester: | Daniel’s Father (Michael) |
| **Test Criteria** | **Michael’s Testing Response** |
| 5 Clickable Menu Buttons | Clicked all 5 buttons, each worked as labeled. “Worked as it should” |
| Player 1 and 2 both respond to their corresponding movement keys | Played as player two, used “Left”, “Right”, “Up” arrow keys to control the yellow player. “Worked well for pure Javascript” |
| The player who is lower on the screen when colliding with their opposition dies and loses a life on the life counter | Collision works, Michael lost all three lives and the life counter incremented. |
| The roof rebounds the player back down the screen | After trying to remain on the roof, player got shot down towards the floor |
| When colliding with the screen boundaries, the player teleports to the opposite side and keeps their momentum | The characters both moved from the left side of the screen to the right and the right to the left |
| After losing 3 lives, the scoreboard appears and displays the lives left as well as the time taken to complete the match | The scoreboard showed up and showed that Michael had 0 lives and lost the game, also the time was displayed |
| High Score screen after the match is completed displays the game that was completed in the quickest time | After Navigating to the high score screen it showed the result of the one game played, so functionality not tested |

* Clear Box Testing

|  |  |  |
| --- | --- | --- |
| Testers: | Michael Mayhew | Daniel Cloran |
| **Test Criteria** | **Expected Outcome** | **Actual Outcome** |
| 5 Clickable Menu Buttons | Five buttons present on the screen each pressable to bring up the corresponding screen | As Planned |
| Movement | Both players move according to WAD and Left, Right, Up arrow keys | As Planned |
| Collision | Collision occurs when the two players collide and the lower player loses a life. | As Planned |
| Rebounding Roof | The roof of the game scene rebounds the player off very quickly | As Planned |
| Boundaries | As the player moves off of the left of the screen, the player then reappears | As Planned |
| Life Counter | After losing 3 lives, the scoreboard appears and displays the lives left as well as the time taken to complete the match | As Planned |
| High Score Screen | High Score screen after the match is completed displays the game that was completed in the quickest time | As Planned |

**Criterion D: Functionality and extensibility of product**  (20 points)

**Functionality of the product:**

Link to video: <https://www.youtube.com/watch?v=ogAzXYF-XFo&feature=youtu.be>

**Extensibility of product:**

The code and elements of the game were turned in a folder submitted in the Drop Box, in which the .html file as well as the elements folder are located in.

**Criterion E: Evaluation** (10 points)

**Evaluation of the product**

The evaluation of the product should refer directly to the success criteria in criterion A, feedback from the client/adviser, as well as any other appropriate feedback obtained. This is different than the testing plan, as this proves the functionality through success criteria

|  |  |  |
| --- | --- | --- |
|  | Description of Success Criteria | Functionality |
| Success Criteria #1 | The fundamental rules of the Arcade Game Joust are present | Every rule from the arcade game, Joust, was implemented and is functional |
| Success Criteria #2 | When a player collides with another player whichever player is lower on the screen dies | The rules behind this success criteria are functional, with the player lower on the screen dying whenever colliding with their opposition. |
| Success Criteria #3 | When a player moves off the left and right of the screen, the player re-appears on the corresponding opposite side | The success criteria is accomplished, as the player moves off the left of the screen the player reappears on the right and likewise on the opposite side. |
| Success Criteria #4 | Two players use 2 unique keyboard button sets to control each ostrich and rider | The success criteria is accomplished, player 1 is controlled by the keys “W”, “A”, and “D” controlling movement and animation, and player two is controlled by the up, left and right arrows accordingly |
| Success Criteria #5 | Title Screen with five buttons, Start, Controls, High Score, Quit | The success criteria is accomplished, with the title screen displaying five separate buttons |
| Success Criteria #6 | Rider dies when colliding with another rider at a lower height | The success criteria is accomplished, with the player dying when they collide with an enemy higher than them. |
| Success Criteria #7 | Display a life counter of 3 lives that increments down to 0 as the players die | The success criteria is accomplished, with the life counter functioning whenever a player is killed, thus incrementing their life counter down one life |
| Success Criteria #8 | Designed player models move upwards when the movement key is pressed | The success criteria is accomplished, with the player models responding to their corresponding movement keys. |
| Success Criteria #9 | Designed player models look to the left and right when the related key is pressed | The success criteria is accomplished, with the player models responding to their corresponding movement keys. |
| Success Criteria #10 | A Player wins when the other player runs out of lives | The success criteria is accomplished, with a player winning when the other player runs out of lives |
| Success Criteria #11 | Two scoreboards are present, the scoreboard after each round showing the time the round took and each life counter, and a scoreboard for the high score showing the quickest round | The success criteria is accomplished, with both scoreboards appearing in their respective areas, as well as the high score scoreboard showing the fastest game |
| Success Criteria #12 | A sound plays each time a player dies | The success criteria is functional, with a sound playing when the player dies |

**Recommendations for the future development of the product**

The student will use the feedback and the evaluation of the specific performance criteria to recommend possible future developments to the product. These recommendations should explain the benefits of these developments.

* Possible improvements
  + Add a countdown to begin each life/game
  + Make the lava on the left and right of the platform kill the player
  + Add more maps
  + Add multiple game modes, in which players can faceoff against automated birds that fly across the screen
    - Add scoring system in which as birds are killed, the player receives points
  + Cleaner interface, with an animated title screen