Kelly Kuhn
N220
July 21, 2023

MidTerm

https://rkkuhn.github.io/N220Spring2023/
I have added my code to the landing page under Mid Term

Algorithm 1: Create a game of Space Invaders using P5.js

Directions: 3+ rows of aliens, moving downwards + back and forth, speeding up over time. Player ship that moves left and right

Player can shoot bullets every .5 seconds when a key is depressed
If a player bullet hits an alien, play a small animation and remove it from the alien grid. Add one to player score.
Display player score (1 point per alien)

Aliens shoot bullets at the player randomly.
If the player gets hit by a bullet or an alien makes it past the player's Y position, stop the game and show "game over" on the screen.
If all the aliens are killed, respawn the whole grid.

Intended Outcome: This is an example of an old game from Atari. The concept is to have a spaceship move back and forth as aliens descend from the top of the screen. The catch here is ensuring the spacecraft can only fire every .5 seconds.

1. Your best breakdown of the problems in this project.
    a. The first is to create global variables.
    b. The second is to set up two classes, one for the player and one for the aliens.
        i. For the player, create a constructor first, movement control using the left and right arrow keys.
        ii. Display the player ship using the constructor variables.
        iii. The Alien creates a constructor, creates the alien movement, and bounces off the left and right edges.
        iv. When an alien ship hits an edge/end, it will move down a line.
        v. Write the code to display the aliens.
    c. Create the function draw area.
        i. Background is black
        ii. If a statement to determine if the game is over or not

iii. If the game is not over, the code will run through the if (!isGameOver) statement.
iv. If the game is over, then I have set up an else statement this goes to displlayGameOver
v. The code will display the score after each loop through the if statement.

d. Create the function (move)
i. Above we called out the player's movement using the arrow keys.
ii. The command is keyIsDown(Left_Arrow) or (Right Arrow)

e. From the options prior, the program now uses two display functions.
i. One for the aliens.
ii. One for the spaceship.

f. The following section creates the bullets.
i. First, we need to make sure the game is not over
ii. If it is not, then the alien bullets will be dropped randomly.
iii. Player bullets are created by using the space bar.

g. To determine if the bullets from the aliens hit their target or if the player hit an alien.
h. Create the checkGameOver function and break if the game is over add a statement on the screen.
i. Display the score.
j. Finally, initialize the aliens.

2. A list of resources you have identified to solve the problems you have identified.
a. Using classes
b. Using several functions to initial different parts of the code
c. Let the command to create global variables
d. Constructor to make it easier to call items from the constructor instead of typing extra code to get the same results.
e. Move the command using the keyIsDown command. I referred to a past assignment where I used the command.
f. Draw a function with a black background.
g. Goggled how to display the score.
h. Googled the best way to initializeAliens – it was recommended to us push.

<u>Reflection:</u> Two paragraphs (five-sentence each) reflection on how the resources (online or in the readings) helped or hindered your implementation.

The lectures helped the most. I spend the last several days going over all my notes. Also, I reviewed all the labs, especially paying attention to my comments. Luckily, when I took C++, we had an assignment like this. So, going into this program, I had a general idea of what I needed to complete.

Trying to get the code in JS with the amount of time was crazy. I know the assignment isn't about finishing, but I can say that looking at all the commands we learned, getting it working would have been great. Classes played a big thing in my code. As soon as I read the directions, I

knew I needed two classes, one for the alien and one for the player. I am happy that I reviewed the lecture on using classes and already had experience using them. It made the setup and thought process more manageable.

I needed to use the console.log () to work through the debugging. Index.html is the file I got working. If I try to fire, it locks up. Everything else works. I did this file after I completed the code in index.html. The original one. I only get a black screen. Index2.html is a breakdown of my trying to get the indexworking.html to work with JS. I needed more time for this section. I waited too long to step back and try this again.

2) A paragraph on how you will approach the problem differently next time.
I would approach this like I should have worked one section at a time. Instead, I thought this was a piece of cake because I programmed this a while back in C++. Rule number one as a programmer is to complete only some things at once. Get sections working, and then move on. The other thing I would do is check and make sure I have my sudo code logic in the correct order. I didn't feel I had enough time to complete all the research. I know I can get this working; I need to step back, refocus my mind on something else, and then return to it.

Sudo Code aka Breakdown:

```
// Author: Kelly Kuhn
// Date: 7-21-23

/*
The problem:

Create a game of space invaders using P5.js.

Requirements:

3+ rows of aliens, moving downwards + back and forth, speeding up over time.
Player ship that moves left and right

Player can shoot bullets every .5 seconds when a key is depressed
If a player bullet hits an alien, play a small animation and remove it from the alien
grid. Add one to player score.
Display player score (1 point per alien)

Aliens shoot bullets at the player randomly.
If the player gets hit by a bullet or an alien makes it past the player's Y position,
stop the game and show "game over" on the screen.
If all the aliens are killed, respawn the whole grid.
*/
```

```
// Global Variables
let player; // instance for player class displayed by the space ship
let aliens = []; // an array to store instances in the Alien Class
let bullets = []; // an array to store instance of the Bullet Class
let alienBulletProbability = 0.005; // Probability value representing the
                                    // chance of aliens firing bullets
let score = 0; // integer => players score
let isGameOver = false; // a boolean to track if the game is over or not

// Create the canvas
function setup(){
   createCanvas(1000, 1000);
}
class Player{
    constructor(){
        this.x = width / 2;
        this.y = height - 50;
        this.w = 50;
        this.h = 20;
        this.speed = 5;
    }

    move(){
      // Movement control using left and right arrow keys
        if (keyIsDown(LEFT_ARROW) && this.x > 0){
            this.x -= this.speed;
        }
        if (keyIsDown(RIGHT_ARROW) && this.x + this.w < width) {
            this.x += this.speed;
        }
    }

    display(){
      // Drawing the player's spaceship
        fill(0, 255, 0);
        rect(this.x, this.y, this.w, this.h);
    }
  }

  // Class Alien
class Alien{
    constructor(x, y){
        this.x = x;
        this.y = y;
        this.r = 20;
        this.xDir = 1;
        this.speed = 1;
    }
```

```
  move(){
    // Horizontal movement and bouncing off the edges
    this.x += this.speed * this.xDir;
      if (this.x + this.r >= width || this.x - this.r <= 0){
          this.xDir *= -1;
      // Move down when hitting the edge
          this.y += 20;
      }
  }

  display(){
    // Drawing the aliens
      fill(255, 0, 0);
      ellipse(this.x, this.y, this.r * 2);
  }
}

function draw(){
  // Game loop: runs continuously and handles the game play
  background(0);

  if (!isGameOver){
      // Game play logic when the game is not over depending on the location
      // of the alien
      player.move();
      player.display();

  for (let alien of aliens){
      alien.move();
      alien.display();
  }

  }
      else{
    // Game play logic when the game is over
          displayGameOver();
  }

  displayScore();
}


// This method is called to handle player movement using the keyIsDown
// command.
function move(){
  // Movement control using left and right arrow keys
  if (keyIsDown(LEFT_ARROW) && this.x > 0){
```

```
        this.x -= this.speed;
    }
    if (keyIsDown(RIGHT_ARROW) && this.x + this.w < width){
        this.x += this.speed;
    }
  }

  // Another method calling the player's ship and the alien method to draw
  // them on the canvas
  function display(){
    // Drawing the player's spaceship
    fill(0, 255, 0);
    // player is a rectangle
    rect(this.x, this.y, this.w, this.h);
  }
  function display(){
    // Drawing the aliens
    fill(255, 0, 0);
    // Aliens are a circle/ellipse
    ellipse(this.x, this.y, this.r * 2);
  }


  // Player Bullets function. If the game is not over a new bullet is created
  // and added to the array bullets
  function keyPressed(){
    if (key === " "){
        if (!isGameOver){
        // -1 ensures the bullet moves upward toward the alien
            bullets.push(new Bullet(player.x, player.y, -1));
      }
    }
  }

  // Alien Bullets random drops down toward the player
if (random() < alienBulletProbability){
    const randomAlien = random(aliens);
    // bullets.push method creates a new bullet
    // 1 ensures the bullet moves downward toward the player
    bullets.push(new Bullet(randomAlien.x, randomAlien.y, 1));
}

  // When bullets hit by using a Bullet class
class Bullet{
    display(){
        // Drawing the bullets
        fill(255);
        ellipse(this.x, this.y, 5);
```

```
    }

    hits(target){
      // Check if the bullet has hit the target (player or alien)
        const d = dist(this.x, this.y, target.x, target.y);
            return d < target.r;
    }
}

// Create the function to check if game is over
function checkGameOver(){
    for (let alien of aliens){
        if (alien.y + alien.r >= player.y){
            isGameOver = true;
            break;
        }
    }

    // if the player is hit by a bullet
    for (let bullet of bullets){
        if (bullet.dir === 1 && bullet.hits(player)){
            isGameOver = true;
            break;
        }
    }
  }

// If game is over display the following message
function displayGameOver(){
    fill(255);
    textSize(32);
    textAlign(CENTER, CENTER);
    text("GAME OVER", width / 2, height / 2);
  }

// When the game is over display the players score right top
function displayScore(){
    fill(255);
    textSize(20);
    textAlign(RIGHT, TOP);
    text("Score: " + score, width - 20, 20);
  }
// Respawn aliens using initializeAliens function
  function initializeAliens(){
    const rows = 3;
    const cols = 10;
    const spacingX = 70;
    const spacingY = 50;
```

```
    for (let i = 0; i < rows; i++){
        for (let j = 0; j < cols; j++){
            aliens.push(new Alien(j * spacingX + 100, i * spacingY + 50));
        }
    }
}
```