Kenza Aboulhouda
Mr. Ettlin
AP Computer Science-Principles, P.1
22 October 2018

Paddle Game Questions

1. Provide a written response for your video that:
    - identifies the programming language;
    - identifies the purpose of your program; and
    - explains what the video illustrates.

    The programming language used for the code in this project is javascript. The purpose of this program was to create a game with a paddle and any amount of balls where if the ball touches the top of the paddle you gain a point and when the ball touches the bottom of the paddle the game restarts from the beginning. This game was made to be not too hard but not too easy, finding that point of captivation from the user playing the game. The video accompanied explains what it illustrates within the video.

2. 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.

    At the beginning of the creation of this program, we went back to former projects that had similar objectives and used those as the base of our code for this program. Examples of such programs include the Acceleration lab, and the Collision lab, both with integral parts of programming that would be very useful in the PaddleGame program. From that point, we went into that former code and began tweaking it to fit the specifications of the PaddleGame project, the majority of this was done independently, with with some parts done collaboratively. The largest difficulty is creating this program was first starting it up and knowing what code to add or remove, and the second, which took up the majority of our time was Collision Detection and the splicing of the balls. Through a collaborative process, with both my partner and other students in the class, we resolved this problem and succeeded in getting to the program to run as specified.

3. Capture and paste the program code segment that implements an algorithm (marked with an oval) that is fundamental for your program to achieve its intended purpose. Your

code segment must include an algorithm that integrates other algorithms and integrates 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. *(Approximately 200 words)*

```
//checking when the ball hits the paddle
  this.checkPaddle = function(){
    //takes location of center of ball
    var distY = abs(this.loc.y - 560)
    //checking if ball is hitting top of paddle
    if((distY < 10) && (this.loc.x > mouseX - 125) && (this.loc.x < mouseX + 125) &&
(this.vel.y > 0)){
      this.sp = 1
    }
    if((distY < 10) && (this.loc.x > mouseX - 125 ) && (this.loc.x < mouseX + 125) &&
(this.vel.y < 0)){
      this.sp = 2
    }
  }
```

This function is fundamental in our program because it allows the balls to be spliced out, or removed, once the ball touches the paddle. Without this function, the game would not work as it was intended to. The independent algorithms in this function each specify different variables and instances, however, when each statement is true, they work in tandem to run the function out splice out the balls. To illustrate, each ball must have this.loc.y = (paddle.y, paddle.length), this.loc.x = (paddle.x, paddle.width), and this.vel is greater than zero (positive) for the ball to spliced out. These algorithms are all independent in that each statement is proven true separately, and in the case that each statement is true, the function runs correctly.

4. Capture and paste the program code segment that contains an abstraction you developed (marked with a rectangle in section 3 below). Your abstraction should integrate mathematical and logical concepts. Explain how your abstraction helped manage the complexity of your program.

```
//function creates the ball
```

```
function Ball(loc, vel, rad, col, sp){
  // Instance variables
  this.loc = loc;
  this.vel = vel;
  this.rad = rad;
  this.col = col;
  this.sp = sp;
}
  //render() draws the ball at the new loc
  this.render = function(){
    fill(this.col);
    ellipse(this.loc.x, this.loc.y, rad, rad);
}
```

The code used in my program uses abstraction to program the balls themselves. For example, in regards to the ellipse function, without abstraction, one would need to create the ellipse and all its variables for it to be shown on the template. Abstraction allows us to just write ellipse(var1, var2, etc..) and it is drawn for us. Abstraction reduces the complexity and length of my program, making all the information easier to write, comprehend, and organize.

5. Capture and paste your entire program code in this section.
    - Mark with an oval the segment of program code that implements the algorithm 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 citations for program code that has been written by someone else.

```
/*
**  Sketch Function
**  Kenza Aboulhouda
**  October 4, 2018
*/

//Global variables
var Balls = [];
var paddle;
var score = 0;
//setup canvas
function setup(){
  var cnv = createCanvas(800, 800);
  cnv.position((windowWidth-width)/2, 30);
```

```
  background(20, 20, 20);
  //# of boids loaded
  numBalls = 10;
  loadBalls(numBalls);
  //
  //creating the lerping paddle
  //
  var loc = createVector(400, 550)
  var vel = createVector(0, 0);
  var width = 250;
  var length = 20;
  var col = color(random(0, 255), random(0, 255), random(0, 255))
  paddle = new Paddle(loc, vel, width, length, col);
}
//load balls
function loadBalls(numBalls){
  for(var i = 0; i < numBalls; i++){
    //what ball needs to know about itself
    var loc = createVector(random(100, 600), 20);
    var vel = createVector(random(-5, 5), random(-5, 5));
    var rad = 25
    var col = color(random(0, 255), random(0, 255), random(0, 255));
    var sp = 3
    var b = new Ball(loc, vel, rad, col, sp);
    //add balls to the array
    Balls.push(b);
  }
}


//draw balls and paddle
function draw(){
  background(20, 20, 20, 6000);
  //score
  textSize(32);
  fill(255, 255, 255);
  text(score, 400, 400)
  //remove outlines
  noStroke();
  paddle.run();
  for(var i = 0; i < Balls.length; i++){
    Balls[i].run();
    var aBalls = Balls[i];
    //splice the balls when they touch paddle
    if(aBalls.sp == 1){
```

```
      Balls.splice(i,1);
      //adds to score for every ball
      score = score + 1;
    }
    //reset game/ add new balls when ball hits bottom of paddle
    if(aBalls.sp == 2){
      //amount of balls in next reset
      var numBalls = Balls.length + 5;
      //resets game and gets rid of previous balls
      Balls = []
      loadBalls(numBalls)
      for(var i = 0; i < Balls.length; i++){
        Balls[i].run;
      }
    }
  }
}

/*
**  Paddle Constructor Function
**  Kenza Aboulhouda
**  October 4, 2018
*/

function Paddle(loc, vel, width, length, col){
  // what paddle needs to know about itself
  //instance variables
  this.loc = loc;
  this.vel = vel;
  this.w = width;
  this.l = length;
  this.col = col;
  // calls other functions
  this.run = function(){
    this.checkEdges();
    this.update();
    this.render();
  }
//lerp the paddle
  this.update = function(){
    //make paddle lerp to middle of rectangle instead of corner
    paddleLength = width/2
    this.loc.x = lerp(this.loc.x, mouseX-paddleLength, .15)
  }
  //checkEdges() reverses speed when the rectangle touches an edge
```

```
  this.checkEdges = function(){
    if(this.loc.x < 0) this.vel.x = -this.vel.x;
    if(this.loc.x > width) this.vel.x = -this.vel.x;
    if(this.loc.y < 0) this.vel.y = -this.vel.y;
    if(this.loc.y > height) this.vel.y = -this.vel.y;
  }
//test github
  //render() draws the paddle at new loc
  this.render = function(){
    fill(this.col);
    rect(this.loc.x, this.loc.y, this.w, this.l);
  }
}

/*
**  Ball Constructor Function
**  Kenza Aboulhouda
**  October 4, 2018
*/

//function creates the ball
function Ball(loc, vel, rad, col, sp){
  // Instance variables
  this.loc = loc;
  this.vel = vel;
  this.rad = rad;
  this.col = col;
  this.sp = sp;
  //this function calls other functions
  this.run = function(){
    this.checkEdges();
    this.update();
    this.render();
    this.checkPaddle();
  }
  //This function changes the location of the ball
  //by adding speed to x and y
  this.update = function(){
    this.loc.x = this.loc.x + this.vel.x;
    this.loc.y = this.loc.y + this.vel.y;
  }
  //checkEdges() reverses speed when the ball touches edge of paddle
  this.checkEdges = function(){
    if(this.loc.x < 0) this.vel.x = -this.vel.x;
    if(this.loc.x > width) this.vel.x = -this.vel.x;
```

```
  if(this.loc.y < 0) this.vel.y = -this.vel.y;
  if(this.loc.y > height) this.vel.y = -this.vel.y;
 }

 //render() draws the ball at the new loc
 this.render = function(){
  fill(this.col);
  ellipse(this.loc.x, this.loc.y, rad, rad);
 }

 //checking when the ball hits the paddle
 this.checkPaddle = function(){
  //takes location of center of ball
  var distY = abs(this.loc.y - 560)
  //checking if ball is hitting top of paddle
  if((distY < 10) && (this.loc.x > mouseX - 125) && (this.loc.x < mouseX + 125) && (this.vel.y >
0)){
    this.sp = 1
  }
  if((distY < 10) && (this.loc.x > mouseX - 125 ) && (this.loc.x < mouseX + 125) && (this.vel.y <
0)){
    this.sp = 2
  }
 }
}
```