Project2 Brief & Pseudo-code

Paul Plew

February 16, 2021

0.1 Brief

0.1.1 Introduction

For this project I am going to create a type of game where the user can click to drop sand, or balls onto the canvas using `p5.js`. The sand will be small squares that will drop from the location of the mouse. These squares will be initialized with a size `size = 5`, an initial `velocity = 2`, and if I can implement it an acceleration similar to that of gravity. The sand will fall to the bottom of the canvas and settle.

The project will use the event listeners `mousePressed()`, `mouseClicked()`, and `keyPressed()`. It will also include a custom PShape as required in the rubric. The extras I might add include changing the color of the sand and adding different modes that will affect gravity or collision detection. As a final extra I will make a mode that randomly places sand throughout the screen as a sort of "what am I supposed to do mode."

0.1.2 Risk Assessment

As of right now I don't know the best way to implement the sand. Objects in `p5.js` and `javascript` in general are a little bit of a pain and from the few tests I have done can just stop working entirely. That as well as the option selection might prove difficult, but I think with enough trial and error I will be able to get everything to work. The `mousePressed()`, and `mouseReleased()` event listeners will also give me some trouble. These functions will take the most time to implement with the transformation of the sand being the quick by comparison. I will also have to put a lot of thought into the physics of the sand as it falls with `terminalVelocity`, as well as `accelerationGravity`.

0.2 Pseudocode

0.2.1 Global Variables

All of these are subject to change as my code progresses but this is a good springboard for further variables and value adjustments.

```
let sandSize = 4;
let liveSand = []; // sand that is falling
let deadSand = []; // sand that has come to rest
let initialVelocity = 2; // initial velocity
let terminalVelocity = 5; // the maximum velocity
let accelerationGravity = 0.1; // gravity
```

0.2.2 setup()

Setup will have a canvas of size: `CreateCanvas(windowWidth, windowHeight);`. It will also make `noFill()` for the shapes. I have yet to decide on a color for the squares as this will most likely be dynamically adjusted with buttons.

```
CreateCanvas(windowWidth, windowHeight);
noFill();
frameRate(20);
```

0.2.3 draw()

draw will start with `background('#000000');` then will call a `updateSand()` function. that will translate the canvas for each of the pieces of sand.

```
background('#000000'); // black for now
updateSand();
drawSand();
```

0.2.4 updateSand()

sandHelper will push, translate and draw a rectangle for each of the items in `liveSand` , but the items in `deadSand` [] will remain unchanged. The translation will be dependant on the current acceleration and velocity of the sand objects. The acceleration of the sand will continue until it has reached its terminal velocity.

```
FOR the items in liveSand
    IF (translated sand is not colliding with sands in deadSand)
        push();
        TRANSLATE based on liveSand
        fill(color);
        RECT(0, 0, sandSize, sandSize)
        pop();
    ELSE
        deadSand[deadSand.length+1] = liveSand piece
    FI
DONE
```

0.2.5 drawSand()

this function will translate for each of the items in liveSand and then draw a rectangle at `x = 0, y = 0` with a size of `sandSize`

```
FOR the items in liveSand
    push();
    TRANSLATE based on liveSand + initialVelocity
    fill(sandColor);
    RECT(0, 0, sandSize, sandSize)
    pop();
DONE
FOR the items in deadSand
    push();
    TRANSLATE based on deadSand
    fill(gray or white);'
```

```
    rect(0, 0, sandSize, sandSize);
    pop();
DONE
```

0.2.6 resetSand()

the function `resetSand()` will erase all the sand objects in `liveSand` and in `deadSand` and reset them to empty arrays.

```
liveSand = [];
deadSand = [];
```

0.2.7 mousePressed()

create new sand particles for the duration of the press dependent on `millis()`

```
IF (someValue % millis()); THEN
    liveSand[liveSand.length+1] = new Sand Piece at
        MouseX and MosueY;
DONE
```

0.2.8 mouseClicked()

if the mouse is clicked in the reset box or a color then the color will change or the sand will be reset

```
IF mouseX and mouseY are some value; THEN
    resetSand();
ELSE IF mouseX and mouseY are some other value; THEN
    color = some color
ELSE IF mouseX and mouseY are some other value; THEN
    color = some other color
ELSE IF mouseX and mouseY are some other value; THEN
```

```
    color = some third color
FI
```

0.2.9 keyClicked()

if the key pressed is `r` the canvas will reset if it is `up arrow` the sand will drop faster if it is `down arrow` the sand will drop slower

```
IF 'r'
    resetSand();
ELSE IF 'up arrow'
    initialVelociy += 1;
ELSE IF 'down arrow'
    initialVelocity -= 1;
FI
```