Kelly Kuhn
N220
July 11th, 2023

Lab 4 – Loops Exercises

https://rkkuhn.github.io/N220Spring2023/

Algorithm 1 -> Click to Drop

Directions: Build an application where, when a user clicks on the screen, a 'brick' is created at that x and y location, and falls down to the bottom of the screen at a speed of 5 pixels / frame. There can be multiple bricks in this application, so you will need to use two arrays and a loop to draw all of the bricks.

Expected Results: Each brick will drop from the location of the mouse on the screen.

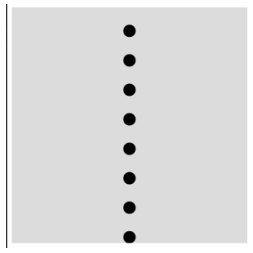
Sudo Code:

```
Set global variable

Create the canvas

Function draw()
    Background (0)

Create the loop to make the bricks
    For Loop
        Update the brick
        Show the update

Create the function for the mouse click
Start the function
    Will need a .push for the brick

Create a brick class
    Create the constructor
    Update the location and speed

Display code using fill and rect
```

Algorithm 2 -> Dripping Water

Directions: Build an application that, every ten frames, draws a circle at the top center of the screen. This circle should move down the screen at 5 pixels / frame. Done properly, there will be multiple circles on the screen at once. Thus, you will need to use an array to store, at a minimum, the Y location of each circle.

Expected Results: A constant flow of circles in the middle all the way down the page.



<u>Sudo Code:</u>

```
Global variable circles

Create function setup for canvas size

Start the draw function and background color

Start the first if statement to see it it’s time to create a new circle

        If (frameCount % 10 === 0)

                Let statement for the new circle providing the x and y requirements.

        For Statement to update and show the circles on the screen

                For (let I = 0; I < circles.length; i++) -> standard for loop setup

                        Create for loop variable for circle = to i

        Have the circle move by 5 px on the y axis

                Circle.y += 5

        Display the circle

                Fill (0)

                Ellipse (circle.x, circle.y, 20)
```

Algorithm 2 -> Neapolitan

<u>Directions:</u> Build an application that, every ten frames, draws a circle at the top center of the screen. This circle should move down the screen at 5 pixels / frame. Done properly, there will be multiple circles on the screen at once. Thus, you will need to use an array to store, at a minimum, the Y location of each circle.

Color Choices:

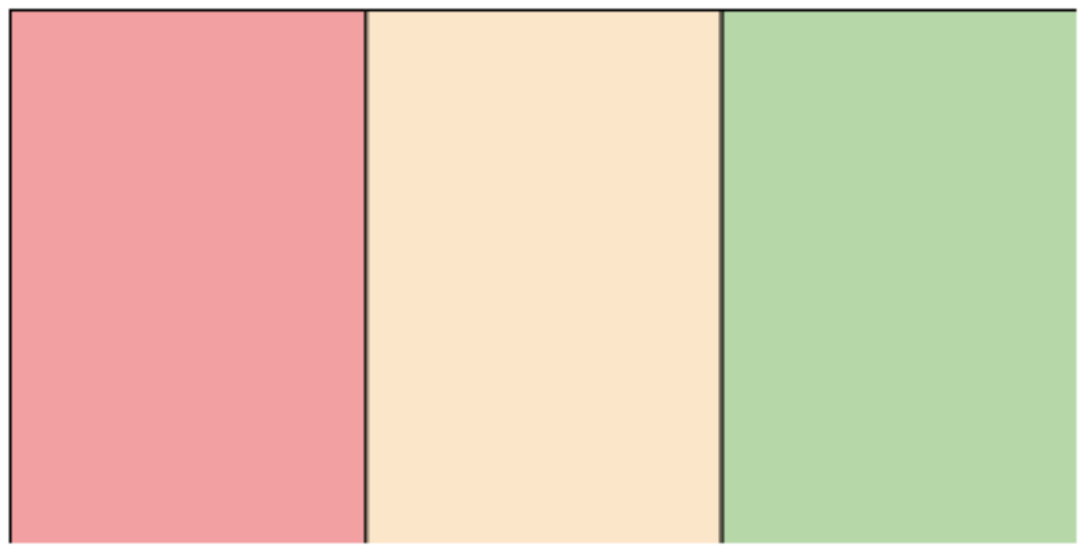
Hex: #FCA0A1
RGBA(252, 160, 161, 1)
HSL(359, 94%, 81%)
CMYB(0%, 37%, 36%, 1%)
Redish orange

Hex: #FCE6C9
RGBA(252, 230, 201, 1)
HSL(34, 89%, 89%)
CMYB(0%, 9%, 20%, 1%)
Cream brown color for the vanilla section

Hex: #B6D7A8
RGBA(182, 215, 168, 1)
HSL(102, 37%, 75%)
CMYB(15%, 0%, 22%, 16%)
Greenish lime color

Please note I am slightly color blink. I can see colors but not variations. This is as close as I could get to the actual colors in Neapolitan ice-cream.

Expected Results:



Sudo Code:

Create the global variable for the colors of the rectangles

```
Start the setup function to create the canvas size

Setup the draw function with the background color

Draw the retangle
    Const rectWidth = width / colors.lenght;
    Const rectHeight = height;

Create a for loop
    Standardy setup let (i=0; i< colors.length; i++)
    Using const x = I * rectWidth creates the variable for x
    Using const y = 0 creates the variable for y

    Color fills of the squares
        Fille (colors[i]; this puts the colors in the correct rectangle
    Color locations for each rectangle using rect
        (x, y, rectWidth, rectHeight)
```

Reflection: This assignment wasn't too bad. I did have to work on Click to Drop. I could not get the program to resent. After some time, I found the code line " new p5()", which did the trick. I am still working on the snake task. I have the code written, but I feel I have the HTML in the app.js, and it should be in the index.HTML file. I am continuing to play around with the code to make it work. I have to say I have a new respect for people on the front end. I admit I used to turn my nose up at front-end stuff. Now I realize there is a lot to making it pretty and user-friendly. I am glad I took this class. The HTML class I had taken only scratched the surface.