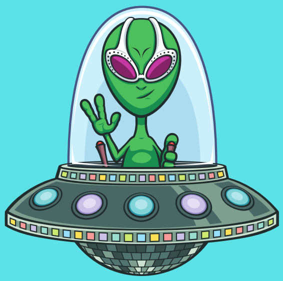
INTRODUCTION TO INTERACTIVE MEDIA

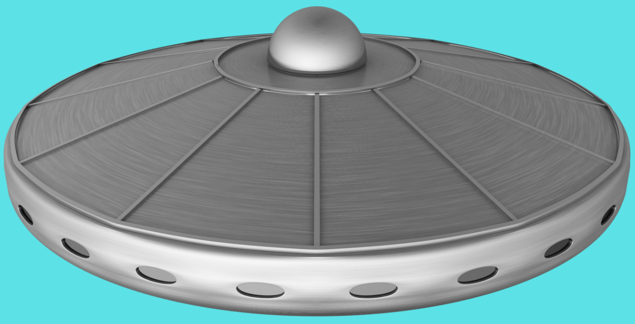
STARSHIP GO DOCUMENTATION

WEEK FIVE - MIDTERM

SYED FAHAD RIZWAN

BACKSTORY





Princess Lyva’s 8-year old daughter, the last descendant of royal blood, has been abducted by aliens from planet Zolo and she must maneuver past Zolo’s fatal landscapes in her spaceship to secure humanity’s future.

CONTROLS

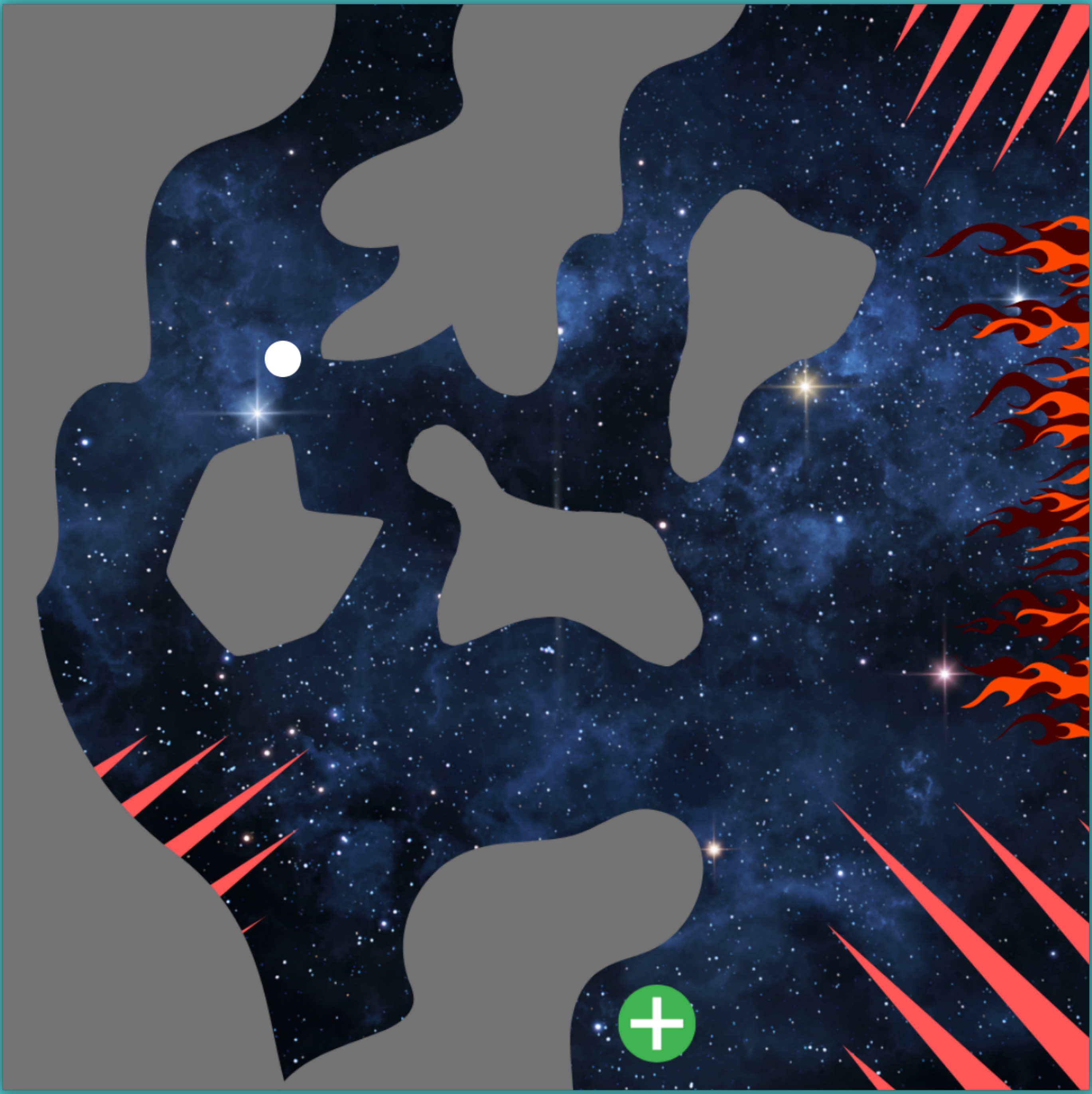
↑ SHIP MOVES UP

↓ SHIP MOVES DOWN

→ SHIP MOVES RIGHT

← SHIP MOVES LEFT

CANVAS



DOCUMENTATION

★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★

PROCESS

01 As a **rudimentary trial maze,** I designed this simple maze via photoshop. I **am not** currently using grids yet keeping them as a last resort

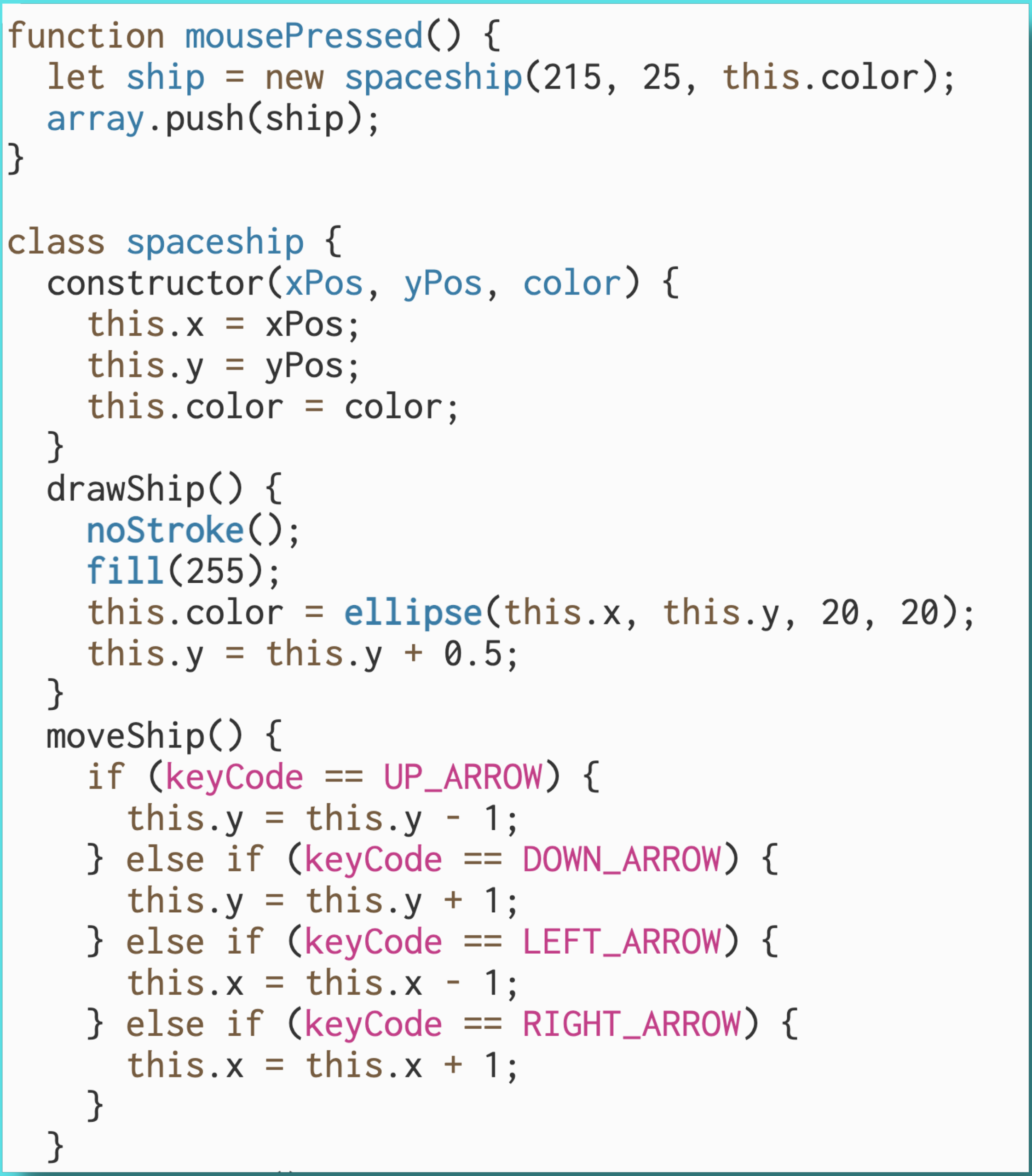
02 I created **monochromatic** spikes, **dichromatic** fire and **monochromatic** walls to make detection easy via **image.get** so I can have an array (snippet below)

03 For now, I'm simply **using an ellipse** as an object for easier control and manipulation. However, I plan to make it cooler once the **concept is operational.**

04 The background image is in the **draw function** to avoid trails once the white sphere moves on screen to move past in the maze.

05 I will work on instruction title page later. However, the **ship (ellipse) summons upon a singular click** anywhere on the canvas

CODE SNIPPET

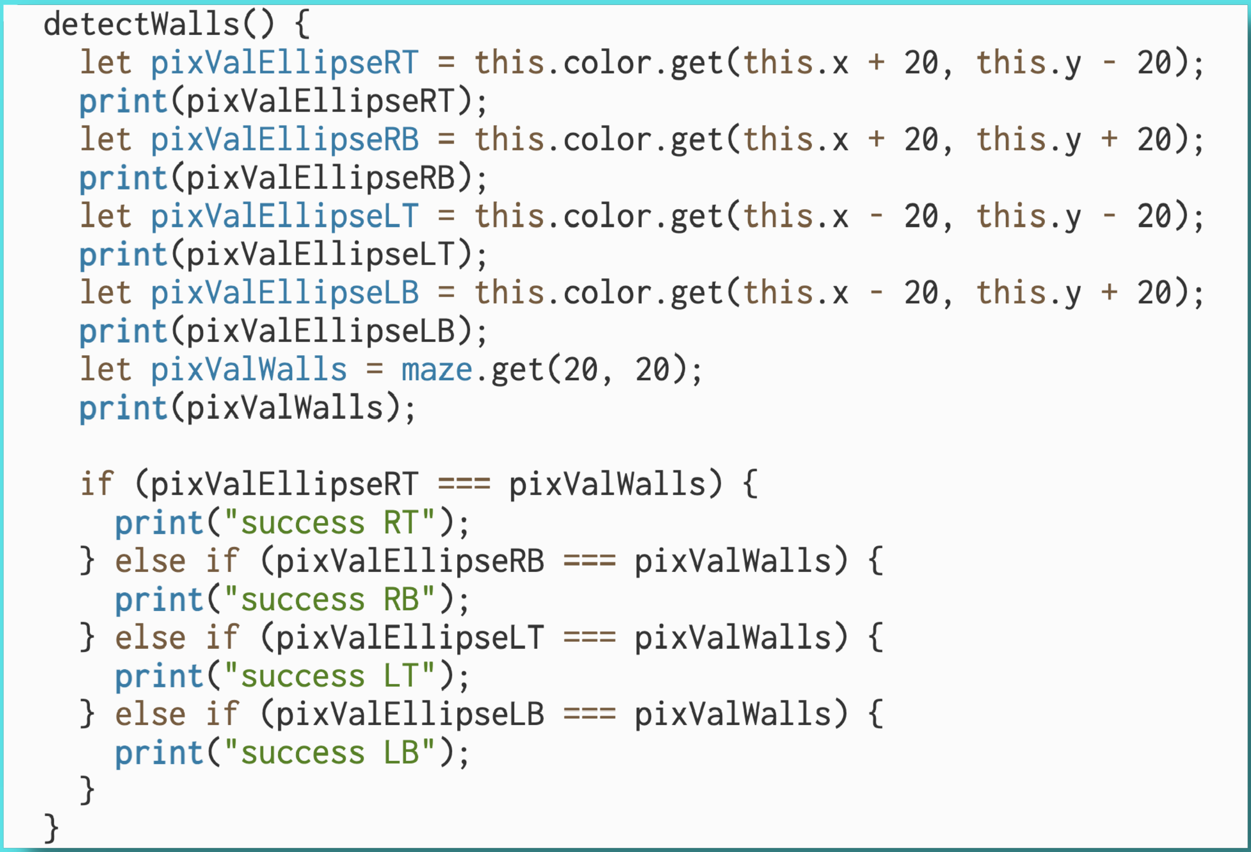


```
function mousePressed() {
  let ship = new spaceship(215, 25, this.color);
  array.push(ship);
}

class spaceship {
  constructor(xPos, yPos, color) {
    this.x = xPos;
    this.y = yPos;
    this.color = color;
  }
  drawShip() {
    noStroke();
    fill(255);
    this.color = ellipse(this.x, this.y, 20, 20);
    this.y = this.y + 0.5;
  }
  moveShip() {
    if (keyCode == UP_ARROW) {
      this.y = this.y - 1;
    } else if (keyCode == DOWN_ARROW) {
      this.y = this.y + 1;
    } else if (keyCode == LEFT_ARROW) {
      this.x = this.x - 1;
    } else if (keyCode == RIGHT_ARROW) {
      this.x = this.x + 1;
    }
  }
```

Class formation, spaceship creation, and movement

CODE SNIPPET



```
detectWalls() {
  let pixValEllipseRT = this.color.get(this.x + 20, this.y - 20);
  print(pixValEllipseRT);
  let pixValEllipseRB = this.color.get(this.x + 20, this.y + 20);
  print(pixValEllipseRB);
  let pixValEllipseLT = this.color.get(this.x - 20, this.y - 20);
  print(pixValEllipseLT);
  let pixValEllipseLB = this.color.get(this.x - 20, this.y + 20);
  print(pixValEllipseLB);
  let pixValWalls = maze.get(20, 20);
  print(pixValWalls);

  if (pixValEllipseRT === pixValWalls) {
    print("success RT");
  } else if (pixValEllipseRB === pixValWalls) {
    print("success RB");
  } else if (pixValEllipseLT === pixValWalls) {
    print("success LT");
  } else if (pixValEllipseLB === pixValWalls) {
    print("success LB");
  }
}
```

This is one **failed attempt** - not sure why - but I think it is theoretically sound. If the coordinates of the edges of my ellipse (spaceship) have the same **pixel value** as that of the walls, I want it to print **success**. If a print, works, so would any manipulation of variables like health and restriction work. I would like to discuss this with you, especially because the print statements on top print arrays that are the **exact same.**

DOCUMENTATION

HOW TO WIN?

Maneuver the spaceship past the spiked walls in a maze configuration until the end line

MAIN VARIABLES

- Spaceship Health
- Spaceship Fuel
- Width
- Height
- Gravity

VARIABLE MANIPULATION

- Upon collision with small spikes, 1/5 health decreases
- Upon collision with **big spikes,** spaceship gets destroyed
- Spaceship fuel constantly decreases
- Height decreases at a constantly increasing rate; until nullified by up command

DOCUMENTATION

RESTOCKING

- Fuel packs at checkpoints
- Health packs at checkpoints

QUESTIONS

- Regarding **object movement,** can I program **two keys at once** to program a specific outcome. For example, pressing **top and left** key at once to move it top-left diagonally?

- What is the **error** in my spike/wall/restock detection formula? I believe the execution of the entire idea is based on this code snippet.

At last, **thank you so much** for the office hours/email help. It's been incredibly helpful to get this far.