MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII

AL REPUBLICII MOLDOVA Universitatea Tehnică a Moldovei Facultatea Calculatoare, Informatică şi Microelectronică Departamentul Inginerie Software și Automatică

Copta Adrian | FAF-223

Report

Laboratory work n.4 part 2

of Computer Graphics

Checked by:

Olga Grosu, *university assistant*

DISA, FCIM, UTM

Chișinău – 2023

1. **Purpose of the task work:**

To learn the behavior of random distribution and vectors.

2. **Condition:**

A: Do the sketch using the function:

- randomGaussian()
- randomSeed()
- random()
- noiseDetail()
- noiseSeed()
- noise()
- map()

B: Vectors project:

Develop a set of rules for simulating the real-world behavior of a creature, such as a nervous fly, swimming fish, hopping bunny, slithering snake, etc. Can you control the object's motion by only manipulating the acceleration? Try to give the creature a personality through its behavior (rather than through its visual design)

3. Program code:

Task A:

```
/*
Adrian Copta | FAF-223
*/

float xOff = 0.0;
float yOff = 0.0;
float increment = 0.02;

void setup() {
  size(400, 400);
  background(255);
  noiseSeed(1); // Set a specific seed for the noise
   noiseDetail(4); // Set noise detail to control the
complexity
}

void draw() {
  loadPixels();

  for (int x = 0; x < width; x++) {
    xOff += increment;
    yOff = 0.0;

    for (int y = 0; y < height; y++) {
      yOff += increment;
        float n = noise(xOff, yOff); // Generate Perlin
noise value

      // Map the noise value to a grayscale color
      int bright = int(map(n, 0, 1, 0, 255));

      // Set pixel color based on the mapped value
      pixels[x + y * width] = color(bright);
    }
  }

  updatePixels();
}
```

Task B:

```
/*
Adrian Copta | FAF-223
*/

PImage water; // Background image of water

Animal[] animals; // Array of animal objects
int numAnimals = 15; // Number of fish

void setup() {
  size(400, 400);
  animals = new Animal[numAnimals];

  for (int i = 0; i < numAnimals; i++) {
    // Initialize fish at random positions
    animals[i] = new Animal(random(width),
random(height));
  }


// Load the background image
  water = loadImage("water.jpg");
  water.resize(width, height);

}

void draw() {
  background(water);

  // Array of fish objects using Perlin noise
  for (int i = 0; i < numAnimals; i++) {
    animals[i].move();
    animals[i].display();
  }
}

class Animal {
  float x, y;

  Animal(float x, float y) {
    this.x = x;
    this.y = y;
  }

  void move() {
    // Move the fish object upwards
    y -= 1;

    // Checking if it has reached the top, then reset
its position to the bottom
    if (y < 0) {
      y = height;
    }

    // Applying Perlin noise to the x-coordinate for
random horizontal movement
    x += noise(y * 0.01) * 2 - 1;
  }

  void display() {
    // Createing the fish using 2D primitives
      fill(random(255), random(255), random(255));
    triangle(x, y, x + 20, y, x + 10, y - 30);
    rect(x + 5, y - 30, 10, 30, 10, 10, 0, 0);
    ellipse(x + 10, y - 45, 10, 15);
    ellipse(x + 10, y - 25, 15, 35);
  }
}
```

4. Program execution:

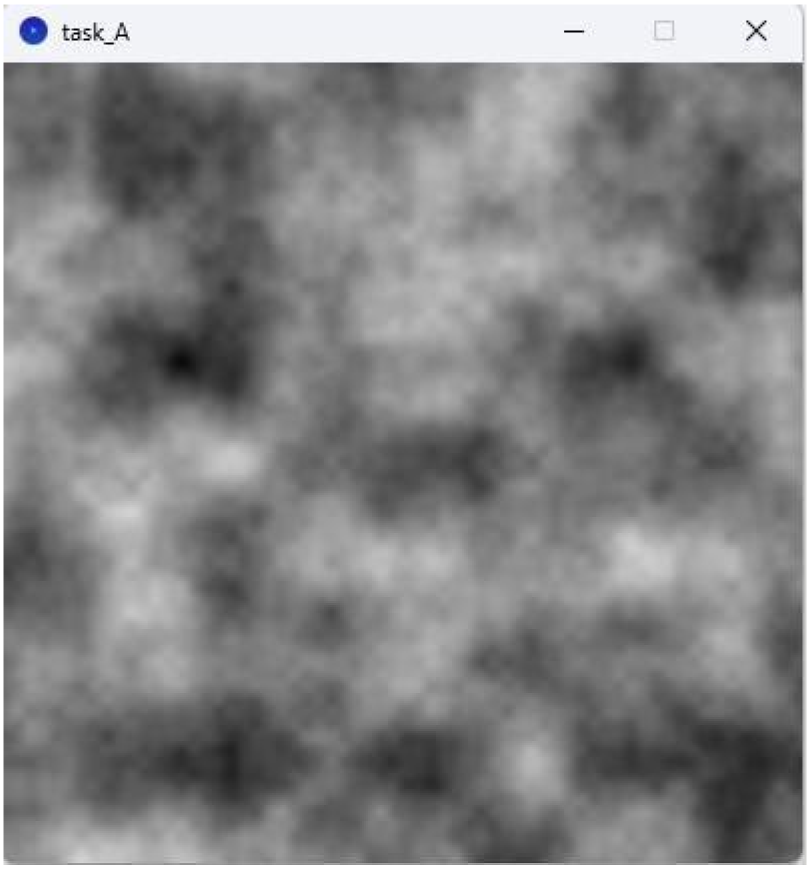

fig.(1) Task A

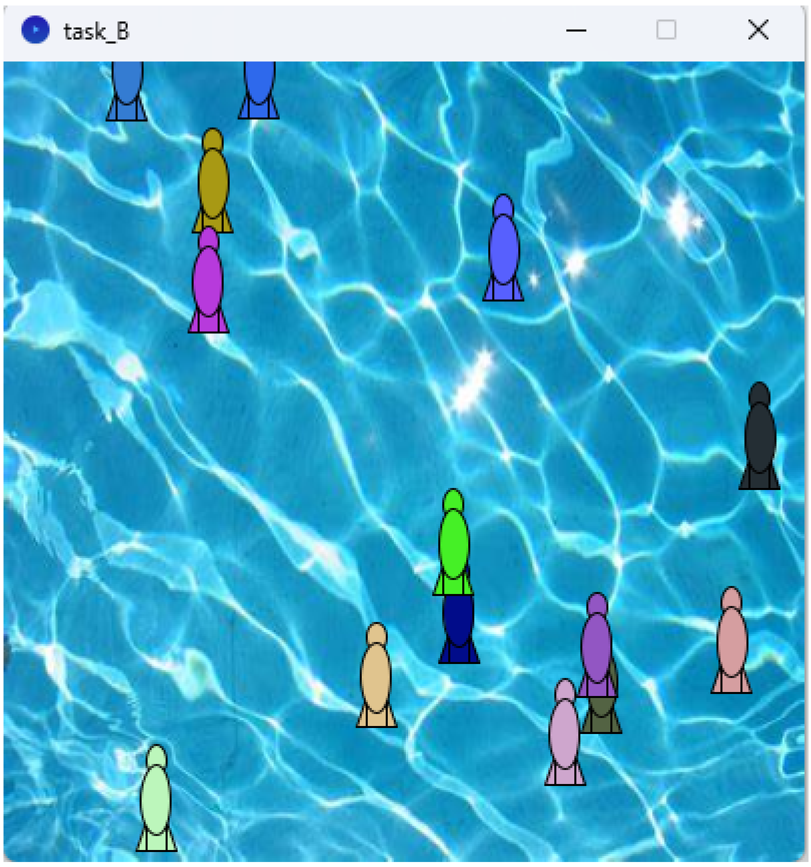

fig.(2) Task B

5. Conclusion:

The code in task A demonstrates the use of Perlin noise in creative coding to produce dynamic, visually appealing patterns. The evolving grayscale image is created through a series of nested loops, with each pixel's color determined by Perlin noise values. By adjusting the noise parameters and experimenting with different mapping functions, the code can be used as a foundation for generating various complex, natural-looking textures and visual effects in creative applications and generative art.

In task B, the goal was to simulate the real-world behavior of a creature using vectors, specifically by manipulating acceleration to control motion. The chosen creature for simulation was a fish, and the implementation successfully achieved the objective of giving it a personality through its behavior rather than visual design. The sketch utilizes an object-oriented approach with a class representing the fish and an array of these objects to simulate a school of fish. The fish exhibit upward motion, giving the illusion of swimming towards the water surface. To add variability to their horizontal movement, Perlin noise is applied to the x-coordinate, resulting in a more natural and unpredictable swimming pattern.

This laboratory project successfully demonstrates the integration of various programming concepts in a visual and interactive context. By simulating an aquatic environment with different entities and user interaction, it highlights the potential of creative coding in educational and entertainment applications. The project encourages exploration and experimentation with Processing for creating dynamic and visually engaging simulations. In summary, this laboratory project not only showcases the power of creative coding but also offers a fun and educational platform for exploring the dynamic behavior of elements within a virtual aquatic world.