MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII AL REPUBLICII MOLDOVA

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică şi Microelectronică Departamentul Inginerie Software și Automatică

Cuzmin Simion Faf-221

Report

Laboratory work n.4

of Computer Graphics

Checked by:

Olga Grosu, *university assistant*

DISA, FCIM, UTM

Chișinău – 2023

Task A

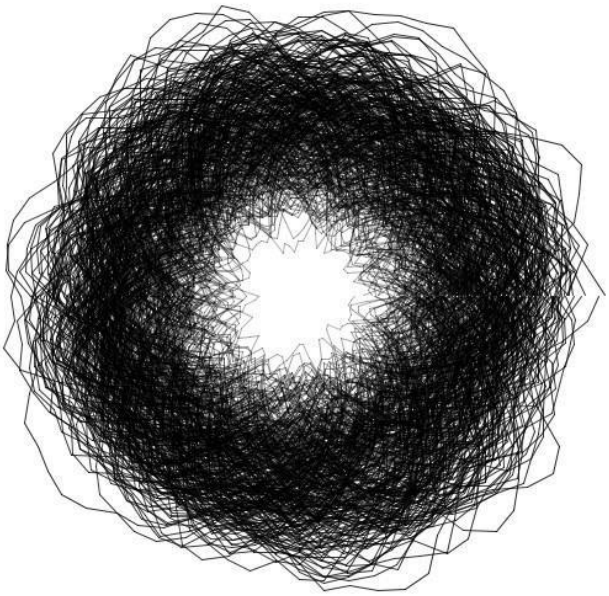
- Do the sketch using the function: • randomGaussian() • randomSeed() • random() • noiseDetail() • noiseSeed() • noise() • map()

- The program code with relevant comments:

```
float xOff = 0;
float yOff =
1000; float
stepSize = 10;
float maxRadius
= 300;

void setup()
{   size(800,
800);
background(25
5);
noFill();
stroke(0,
50);
randomSeed(42
);
noiseSeed(42)
;
noiseDetail(4
, 0.5);
}
```

```
void draw()
{   float
xPrev = -1;
float yPrev
= -1;

  for (float a = 0; a < TWO_PI; a
+= radians(5)) {     float rad =
maxRadius * noise(xOff, yOff);
float x = width / 2 + cos(a) *
rad;
    float y = height / 2 + sin(a) *
rad;

    if (xPrev > 0 && yPrev > 0) {
      float alpha = map(rad, 0,
maxRadius, 50, 255);       stroke(0,
alpha);
      line(xPrev, yPrev, x, y);
    }

    xPrev =
x;     yPrev
= y;

    xOff += 0.1;
  }

  yOff += 0.01;

  if (frameCount == 400)
{     noLoop();
  }
}
```

- Screen printing of program execution:



Task 2

- Make sketch with 2d primitives function and move it ,you can use random function or increase the coordonate of your sketch combination), use VARIABLES, CONDITIONALS, LOOPS function and create your own function and call it in Draw function

- The program code with relevant comments:

```
PVector[] fishPositions;
PVector[] fishVelocities;
```

```
void setup() {   size(800, 600);
background(0, 100, 200); // Deep
blue background

  fishPositions = new
PVector[3];   fishVelocities =
new PVector[3];

  // Initialize fish positions and velocities   for
(int i = 0; i < fishPositions.length; i++)
{     fishPositions[i] = new PVector(random(width),
random(300, height - 100));     fishVelocities[i] =
new PVector(0, 0);
  }

  // Draw static elements - sand, algae,
stars, and other creatures
drawStaticElements();
}

void
drawStaticElements()
{   drawSand();
drawAlgae();
drawMarineStars(1);
drawOtherMarineCreatu
res(5);
```

```
}

void drawSand()
{   noStroke();
fill(255, 235, 170);
// Sandy color
  rect(0, height - 100, width, 100); //
Draw sand at the bottom
}

void drawAlgae() {
  for (int i = 0; i < 50; i++) {     float x =
random(width); // Random x position     float y
= height - 100 + random(80); // Random y
position on the sand     float len = random(30,
100); // Random algae length

    // Algae color
(greenish)     fill(0,
255, 100, 150);

    // Draw an algae
strand
beginShape();
curveVertex(x, y);
curveVertex(x, y);
curveVertex(x - 10, y
- len / 3);
```

```
curveVertex(x + 10, y
- len / 2);
curveVertex(x, y -
len);
curveVertex(x, y -
len);     endShape();

    // Draw some algae bubbles      fill(100, 200, 255,
200); // Light blue bubbles      ellipse(x + random(-5,
5), y - len - random(10, 30), random(5, 15), random(5,
15));
 }
}

void drawMarineStars(int
numStars) {   for (int i =
0; i < numStars; i++)
{     float x =
random(width);     float y
= random(200, height -
100);     float starSize =
random(20, 40);

    // Marine star     fill(255,
255, 0);     beginShape();
for (int j = 0; j < 5; j++)
{       float angle = TWO_PI / 5
* j;       float xOffset =
```

```
cos(angle) * starSize;
float yOffset = sin(angle) *
starSize;       vertex(x +
xOffset, y + yOffset);
xOffset = cos(angle + PI/5) *
(starSize * 0.4);       yOffset =
sin(angle + PI/5) * (starSize *
0.4);       vertex(x + xOffset, y
+ yOffset);
    }
    endShape(CLOSE);
  }
}

void drawOtherMarineCreatures(int
numCreatures) { for (int i = 0; i <
numCreatures; i++) {
    float x =
random(width);     float y
= random(200, height -
100);     float
creatureSize = random(20,
60);

    // Customized marine creature
fill(random(255), random(255),
random(255));     beginShape();
for (int j = 0; j < 8; j++)
```

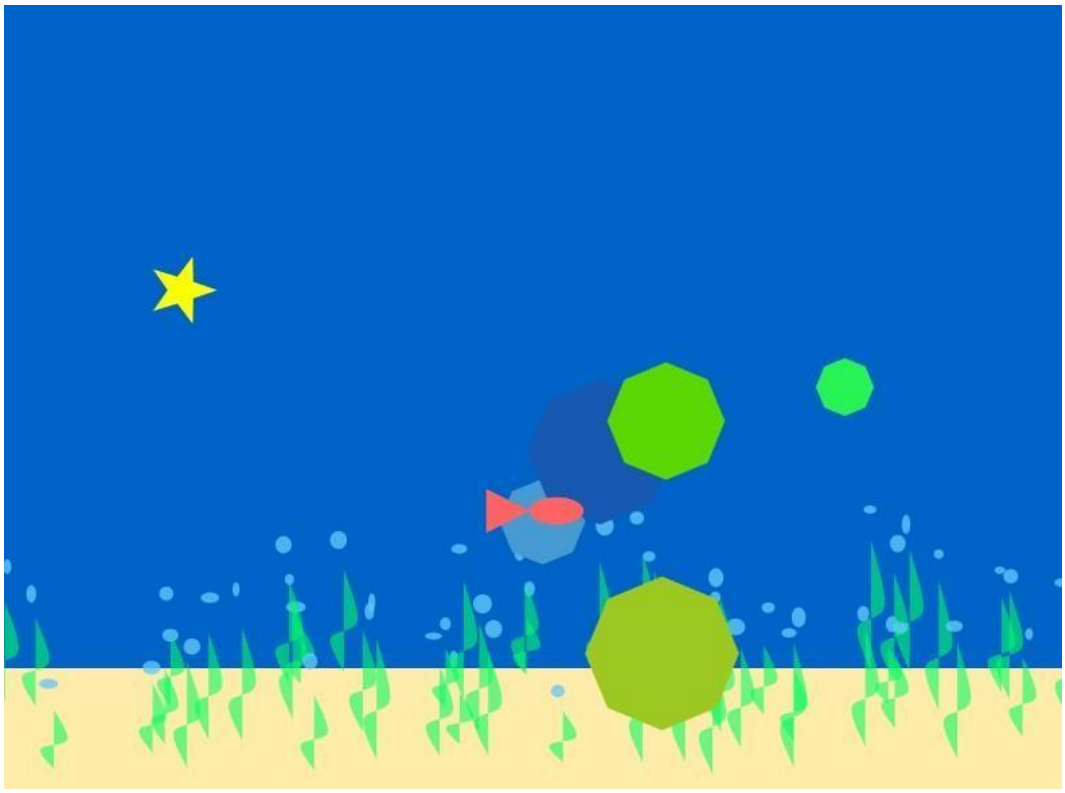
```
{       float angle = TWO_PI / 8
* j;       float xOffset =
cos(angle) * creatureSize;
float yOffset = sin(angle) *
creatureSize;       vertex(x +
xOffset, y + yOffset);
    }
    endShape(CLOSE);
  }
}

void draw() {   background(0, 100,
200); // Refresh the background

  // Draw static elements - sand, algae,
stars, and other creatures
drawStaticElements();

  // Update and draw fish
for (int i = 0; i <
fishPositions.length; i++)
{     PVector target = new
PVector(mouseX, mouseY); //
Cursor position as the target
PVector desired =
PVector.sub(target,
fishPositions[i]);
```

```
    // Check if the right mouse button is
pressed (mouse button 2)     if
(mousePressed && mouseButton == RIGHT) {
      // If the right mouse button is pressed, make the
fish move away from the cursor      desired.mult(-1); //
Reverse the direction
    }

    desired.setMag(2); // Adjust the
speed     fishVelocities[i] = desired;
fishPositions[i].add(fishVelocities[i
]);

    // Draw fish at the updated position
drawFish(fishPositions[i].x,
fishPositions[i].y);
  }
}

void drawFish(float
x, float y) {   float
fishSize = 40;
float tailSize =
fishSize * 0.8;

  // Fish body
fill(255, 100, 100);
ellipse(x, y, fishSize,
fishSize * 0.5);   //
```

```
Fish tail   fill(255,
100, 100);   triangle(x -
fishSize / 2, y, x -
fishSize / 2 - tailSize,
y - tailSize / 2, x -
fishSize / 2 - tailSize,
y + tailSize / 2);
}

void mousePressed() {
  // Check if the right mouse button (mouse
button 2) is pressed   if (mouseButton ==
RIGHT) {
    // When the right mouse button is pressed, make the
fish move away from the cursor
    for (int i = 0; i <
fishPositions.length; i++) {
      PVector target = new PVector(mouseX, mouseY); //
Cursor position as the target
      PVector desired = PVector.sub(target,
fishPositions[i]);       desired.mult(-1); //
Reverse the direction to move away from the cursor
desired.setMag(2); // Adjust the speed
fishVelocities[i] = desired;
    }
  }
}
```

```
void mouseReleased() {
  // Check if the right mouse button (mouse
button 2) is released   if (mouseButton ==
RIGHT) {
    // When the right mouse button is released, make
the fish follow the cursor again     for (int i = 0; i <
fishPositions.length; i++) {
      PVector target = new PVector(mouseX, mouseY); //
Cursor position as the target
      PVector desired =
PVector.sub(target, fishPositions[i]);
desired.setMag(2); // Adjust the speed
fishVelocities[i] = desired;
    }
  }
}
```

- Screen printing of program execution:



- Student's conclusions and reflections:

This code presents an engaging and creative exploration of visual design and interactivity in Processing. It establishes a dynamic underwater scene with an array of elements, including a sandy seafloor, playful algae, radiant marine stars, and imaginative marine creatures. The use of vivid colors, custom shapes, and randomization adds a sense of wonder and charm to the underwater environment. The central feature of the sketch is the school of fish that move in response to the user's cursor. The fish exhibit an intelligent and interactive behavior, following the cursor when the left mouse button is clicked and swimming away from it when the right mouse button is pressed. This interactive behavior gives the fish a sense of curiosity and playfulness, enhancing the overall experience. The code demonstrates the power of vector manipulation to control the motion of objects and provides users with an interactive canvas for creative expression. It is an excellent example of how coding can be used to create artistic and interactive visual experiences, making it a valuable resource for those looking to explore the potential of creative coding in the realm of visual art and interactivity.