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.5

of Computer Graphics

Checked by: [i]

Элементы оглавления не найдены.

Olga Grosu, *university assistant*

DISA, FCIM, UTM

Chișinău – 2023

Task A

- Do the sketch using the function:
  Incorporate the concept of forces into your ecosystem (you can use the ecosystem from Task 4 point B or create something new). Try introducing other elements into the environment (food, a predator) for the creature to interact with. Does the creature experience attraction or repulsion to things in its world? Can you think more abstractly and design forces based on the creature's desires or goals?

- The program code with relevant comments:

```
import
java.util.ArrayList;
import
java.util.Iterator;

ArrayList<Fish> fishes;
ArrayList<Food> foods;
ArrayList<Shark> sharks;
ArrayList<Algae> algae;
ArrayList<Bubble> bubbles;
ArrayList<Star> stars;
```

```
 }
void setup()
{   size(800,
600);
background(0,
100, 200);

  fishes = new
ArrayList<Fish>();   foods
= new ArrayList<Food>();
sharks = new
ArrayList<Shark>();
algae = new
ArrayList<Algae>();
bubbles = new
ArrayList<Bubble>();
stars = new
ArrayList<Star>();

  // Initialize fish
positions and velocities
for (int i = 0; i < 5; i++)
{ fishes.add(new
Fish(random(width),
random(300, height - 100)));


  // Initialize background elements
```

```
  for (int i = 0; i < 20; i++) {
    algae.add(new Algae(random(width),
random(height - 100)));     bubbles.add(new
Bubble(random(width), random(height - 100)));
stars.add(new Star(random(width), random(200,
height - 100)));
  }
}

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

  // Draw background
elements   for (Algae
a : algae)
{     a.display();
  }

  for (Bubble b : bubbles)
{     b.display();
  }

  for (Star s :
stars)
{     s.display();
  }
```

```
 }
  // Draw food
for (Food food :
foods)
{     food.displa
y();
  }

  // Update and
draw fish   for
(Fish fish :
fishes)
{     fish.update()
;
fish.display();

    // Check if fish reached any food
    Iterator<Food> foodIterator =
foods.iterator();
  while
(foodIterator.hasNext())
{       Food food =
foodIterator.next();
      float d = dist(fish.position.x,
fish.position.y, food.position.x,
food.position.y);       if (d < 15) {
        // If fish reached the food, remove the
food         foodIterator.remove();
```

```
      }
    }
  }

  // Update and draw sharks
  Iterator<Shark> sharkIterator =
sharks.iterator();   while
(sharkIterator.hasNext())
{     Shark shark =
sharkIterator.next();
shark.update();
shark.display();

    // Check if shark caught any fish
    Iterator<Fish> fishIterator
= fishes.iterator();     while
(fishIterator.hasNext())
{       Fish fish =
fishIterator.next();
      float d = dist(shark.position.x,
shark.position.y, fish.position.x,
fish.position.y);       if (d < 20) {
        // If shark caught the fish,
remove both shark and fish
sharkIterator.remove();
fishIterator.remove();
      }
```

```
 }
    }
  }
}

void mousePressed()
{   if (mouseButton
== RIGHT) {
    // Add shark at the right mouse click
position     sharks.add(new Shark(new
PVector(mouseX, mouseY)));
  } else {
    // Add food at the mouse click position
 foods.add(new Food(new PVector(mouseX,
                                mouseY)));

}

class Algae
{   float x,
y;

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

```
  void   display()
{        //  Algae
color   (greenish)
fill(0,  255,  100,
150);

    // Draw an algae
strand
beginShape();
curveVertex(x, y);
curveVertex(x, y);
curveVertex(x - 10, y
- 20);
curveVertex(x + 10, y
- 40);
curveVertex(x, y -
80);
curveVertex(x, y -
80);     endShape();
  }
}

class
Bubble
{    float
x,      y;
```

```
 }
float
radius;

  Bubble(float x, float y) {
    this.x = x;
this.y = y;
this.radius =
random(5, 15);
  }

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

class
Star
{    flo
at x, y;
float
size;

  Star(float x,
float y)
```

```
{     this.x = x;
this.y = y;
this.size =
random(10, 30);
  }

  void display() {     //
Marine star     fill(255, 255,
0);     beginShape();     for
(int j = 0; j < 5; j++)
{       float angle = TWO_PI /
5 * j;       float xOffset =
cos(angle) * size;       float
yOffset = sin(angle) * size;
vertex(x + xOffset, y +
yOffset);       xOffset =
cos(angle + PI/5) * (size *
0.4);       yOffset =
sin(angle + PI/5) * (size *
0.4);       vertex(x +
xOffset, y + yOffset);
    }
    endShape(CLOSE);
  }
}
```

```
 }
class Fish
{   PVector
position;
  PVector velocity;

  Fish(float x, float
y) {     position =
new PVector(x, y);
velocity = new
PVector(0, 0);


  void update()
{     if
(foods.size() >
0) {
      Food nearestFood = findNearestFood();
      PVector desired =
PVector.sub(nearestFood.position, position);
desired.setMag(2);       velocity = desired;
    }

    // Boundary
checks     if
(position.x <
0)
```

```
{       position
.x = 0;
      velocity.x *= -1; // Reverse the x-direction
velocity
    } else if (position.x >
width) {       position.x =
width;
      velocity.x *= -1; // Reverse the x-direction
velocity
    }

    if (position.y <
0) {       position.y
= 0;
      velocity.y *= -1; // Reverse the y-direction
velocity
    } else if (position.y >
height) {       position.y =
height;
      velocity.y *= -1; // Reverse the y-direction
velocity
    }

    position.add(velocity);
  }
```

```
 }

  void
display()
{     // Fish
body
fill(255, 100,
100);
    ellipse(position.x, position.y, 40, 20);

    // Fish
tail
fill(255, 100,
100);
triangle(posit
ion.x - 20,
position.y,
position.x -
30, position.y
- 10,
position.x - 30, position.y + 10);
  }
}

class Food
{   PVector
position;
```

```
  Food(PVector pos)
{     position =
pos;
  }

  void display()
{     // Food
color (yellow)
fill(255, 255,
0);
    ellipse(position.x, position.y, 15, 15);
  }
}

class Shark
{   PVector
position;
  PVector velocity;

  Shark(PVector pos)
{     position = pos;
velocity = new
PVector(0, 0);
  }

  void update()
{     if
```

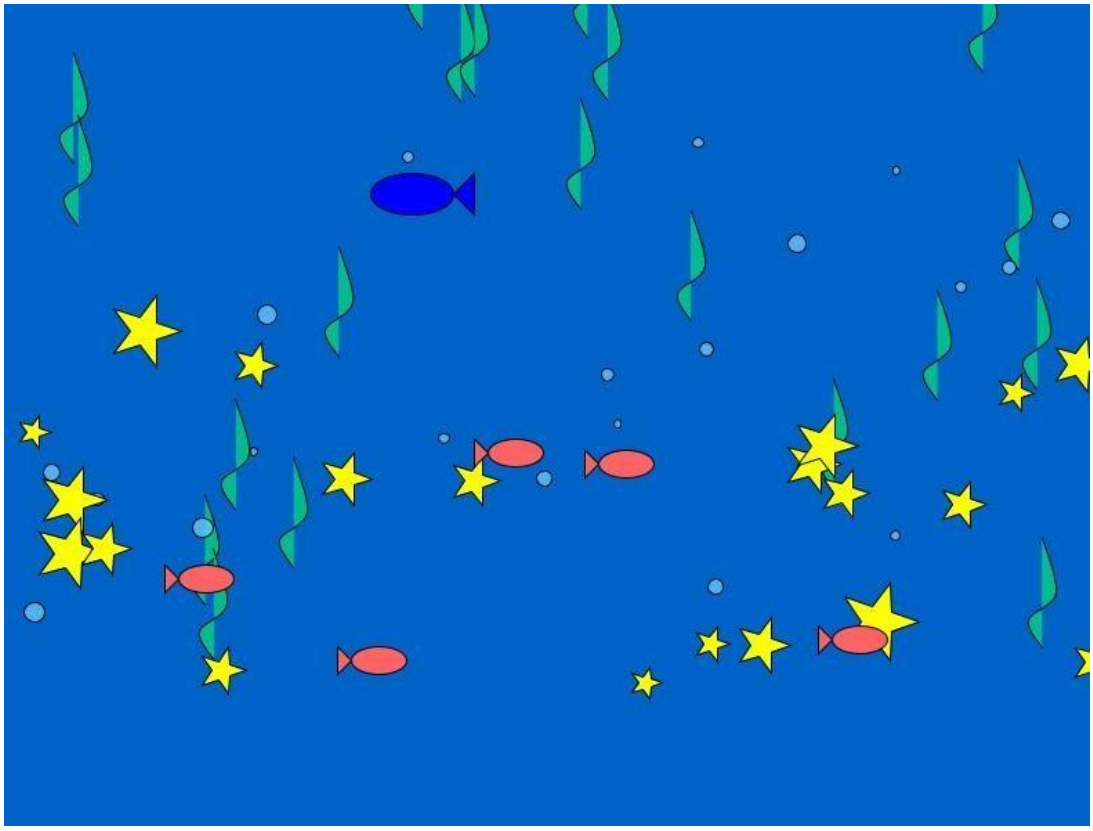
```
 }
(fishes.size() >
0) {
      Fish nearestFish = findNearestFish();
      PVector desired =
PVector.sub(nearestFish.position,
position);       desired.setMag(2);
velocity = desired;
    }

    // Boundary checks
    if (position.x < 0 || position.x > width ||
position.y < 0 || position.y > height) {
```

```
   sharks.remove(this); // Remove the shark if it
goes outside the canvas
    }

    position.add(velocity);
  }

  void
display()
{     //
Shark body
fill(0, 0,
255);
    ellipse(position.x, position.y, 60, 30);

    // Shark
tail
fill(0, 0,
255);
    triangle(position.x + 30, position.y,
position.x + 45, position.y - 15, position.x + 45,
position.y + 15);
  }
}
```

```
Food
findNearestFood()
{   if (foods.size()
> 0) {     Food
nearest =
foods.get(0);
    float record = dist(nearest.position.x,
nearest.position.y, mouseX, mouseY);
    for (Food food : foods) {
      float d = dist(food.position.x,
food.position.y, mouseX, mouseY);       if (d
< record) {         record = d;
nearest = food;
      }
    }
    return
nearest;
} else
{     retur
n null;
  }
}

Fish
findNearestFish()
{   if
(fishes.size() > 0)
```

```
{     Fish nearest =
fishes.get(0);
float record =
dist(nearest.positio
n.x,
nearest.position.y,
mouseX, mouseY);
    for (Fish fish : fishes) {
      float d = dist(fish.position.x,
fish.position.y, mouseX, mouseY);       if (d
< record) {          record = d;
nearest = fish;
      }
    }
    return
nearest;
} else
{     retur
n null;
  }
}
```

- Screen printing of program execution:



- Student's conclusions and reflections:

This code showcases a captivating underwater world, meticulously crafted using the Processing language. The scene comes to life with diverse background elements such as gracefully swaying algae, whimsical bubbles, and sparkling stars. The incorporation of these vivid and dynamic visual elements adds depth and complexity to the underwater ambiance. The main focus of the sketch revolves around a school of fish that respond intelligently to user interactions.

The fish, designed with vibrant colors and distinct shapes, exhibit a sophisticated behavior. Their movement is dictated by the user's cursor, creating an interactive and engaging experience. When the left mouse button is clicked, the fish gracefully swim towards the cursor, showcasing a curious and playful demeanor. Conversely, the fish elegantly navigate away from the cursor when the right mouse button is pressed, evoking a sense of responsiveness and interactivity.

The code effectively employs vector manipulation to control the intricate motion of the fish, showcasing the flexibility and power of creative coding. This piece serves as a compelling example of how programming can be utilized to craft visually stunning and interactive art. It provides a canvas for users to explore their creativity, offering a valuable resource for those eager to delve into the realm of artistic coding and interactive visual experiences.

---

i