****

**MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII**

**AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Informatică şi Ingineria Sistemelor**

Chistol Maxim

Gr.IA-233

**Raport**

**pentru lucrarea de laborator Nr.2**

***la cursul de “Grafica pe Calculator”***

Verificat:

**Mariana Osovschi,** *Lector Universitar,*

Facultatea FCIM,

**Chișinău – 2024**

**Tema:** Studierea primitivelor grafice simple 2D

**Scopul lucrării:**

Implementarea transformărilor grafice asupra unei scene 2D utilizând setul de funcționalități a bibliotecii (JavaScript) p5.js.

**Sarcina lucrării:**

Transformări grafice, folosirea acestor elemente in proiectul nostrum,2D în biblioteca grafică p5js:

• (translate)

• (scale)

• (rotate)

• (shear)

**Codul:**

**let jitter = 0;**

**let angle = 0;**

**let x = 0;**

**let y = 0;**

**let dim = 100.0;**

**let opacity = 0;**

**let scaling = 1;**

**function setup() {**

**createCanvas(800, 400);**

**}**

**// super mario !!!**

**function draw() {**

**background(135,206,250);**

**strokeWeight(0)**

**fill(0,126,0);**

**rect(0, 350, 800, 55)**

**strokeWeight(2)**

**// desenarea elementelor pe scena**

**//decoratia 1**

**strokeWeight(0);**

**ellipse(90,300,50,30);**

**ellipse(90,320,70,40);**

**ellipse(90,340,90,50);**

**//decoratia 2**

**ellipse(587,300,50,30);**

**ellipse(587,320,70,37);**

**ellipse(587,340,90,50);**

**strokeWeight(2)**

**//monedele**

**strokeWeight(0)**

**push();**

**fill(245, 238, 1);**

**let s = abs(cos(frameCount \* 0.007+1.0));**

**scaling = s; // Actualizare scalare**

**translate(300, 300);**

**scale(scaling);**

**ellipse(0, 0, 30, 30); // Coordonatele sunt în raport cu sistemul de coordonate translatat și scalat**

**ellipse(-40, 10, 30, 30);**

**ellipse(-80, 10, 30, 30);**

**ellipse(40, -20, 30, 30);**

**ellipse(80, -30, 30, 30);**

**ellipse(120, -10, 30, 30);**

**ellipse(160, 10, 30, 30);**

**ellipse(200, 10, 30, 30);**

**ellipse(240, 10, 30, 30);**

**fill(39, 197, 39);**

**pop();**

**strokeWeight(2)**

**//norii**

**strokeWeight(0);**

**fill(255,255,255);**

**push();**

**// Animează creșterea valorii x**

**x = x + 0.8;**

**// Dacă forma iese din grilă, resetați poziția**

**if (x > 100 +dim) {**

**x = -dim;**

**}**

**translate(x, 20 / 2 - dim / 2);**

**//norul mare**

**ellipse(70,80,60,40);**

**ellipse(70,100,100,50);**

**ellipse(70,115,60,40);**

**//norii mici**

**ellipse(170,80,50,30);**

**ellipse(190,180,50,30);**

**ellipse(160,270,60,40);**

**ellipse(370,80,50,30);**

**ellipse(300,250,50,30);**

**ellipse(470,80,60,40);**

**ellipse(530,180,50,30);**

**ellipse(500,270,50,30);**

**ellipse(700,200,60,40);**

**pop();**

**strokeWeight(2);**

**//dicoratia**

**fill(202,129,2);**

**rect(200, 130, 55, 55, 5, 5, 5, 5);**

**rect(258, 130, 55, 55, 5, 5, 5, 5);**

**rect(370, 130, 55, 55, 5, 5, 5, 5);**

**fill(4,187,4);**

**rect(350, 295, 45, 55, 5, 5, 5, 5);**

**rect(345, 290, 55, 25, 5, 5, 5, 5);**

**rect(650, 295, 45, 55, 5, 5, 5, 5);**

**rect(645, 280, 55, 25, 5, 5, 5, 5);**

**strokeWeight(2);**

**//mario**

**push();**

**// Animează crse din grilă, resetați poziția**

**if (x > 130 + dim) {**

**x = -dim;**

**}**

**translate(x, 100 / 2 - dim / 2);**

**fill(255,0,0);**

**rect(90,270,40,10,5,5,5,5);**

**fill(248,182,15,);**

**rect(95,280,30,15,5,5,5,5);**

**fill(7,7,162);**

**rect(90,296,40,30,7,7,7,7);**

**rect(113,320,8,30,7,7,0,0);**

**rect(100,320,8,30,7,7,0,0);**

**fill(247,0,0);**

**rect(105,300,35,6,7,7,7,7);**

**strokeWeight(4)**

**point(120,285);**

**pop();**

**//soarle**

**push();**

**if (second() % 2 === 0) {**

**jitter = random(-0.1, 0.1);**

**}**

**angle = angle + jitter;**

**let c = cos(angle);**

**// Mutați forma în centrul grilei**

**translate(1430 / 2, 160 / 2);**

**// Aplicați rotația finală**

**rotate(c);**

**strokeWeight(0);**

**fill(245, 238, 1);**

**circle(0, 0, 110);**

**pop();**

**//personajul suplimentar 1**

**push();**

**if (second() % 2 === 0) {**

**jitter = random(-0.1, 0.1);**

**}**

**angle = angle + jitter;**

**let a = cos(angle);**

**// Mutați forma în centrul grilei**

**translate(1430 / 2, 160 / 2);**

**// Aplicați rotația finală**

**rotate(c);**

**//corpul**

**fill(128, 0, 0);**

**strokeWeight(2);**

**square(520,320,30);**

**//ochii**

**strokeWeight(5);**

**fill(0);**

**arc(530, 327+ 20 / 9, 20 / 6, 20 / 10, 0, PI);**

**strokeWeight(2);**

**//gura**

**fill(170,0,219);**

**rect(527,339,10,5);**

**pop();**

**push();**

**translate(520, 320);**

**rotate(radians(angle));**

**// Corpul**

**fill(128, 0, 0);**

**strokeWeight(2);**

**square(0, 0, 30);**

**// Ochii**

**strokeWeight(5);**

**fill(0);**

**arc(10, 7 + 20 / 9, 20 / 6, 20 / 10, 0, PI);**

**strokeWeight(2);**

**// Gura**

**fill(170, 0, 219);**

**rect(7, 19, 10, 5);**

**pop();**

**angle += 1; // Incrementarea unghiului pentru a obține animația de rotație**

**//marginile**

**strokeWeight(20)**

**line(10,10,10,390);**

**line(10,390,790,390);**

**line(10,10,790,10);**

**line(790,390,790,10);**

**//treunghi**

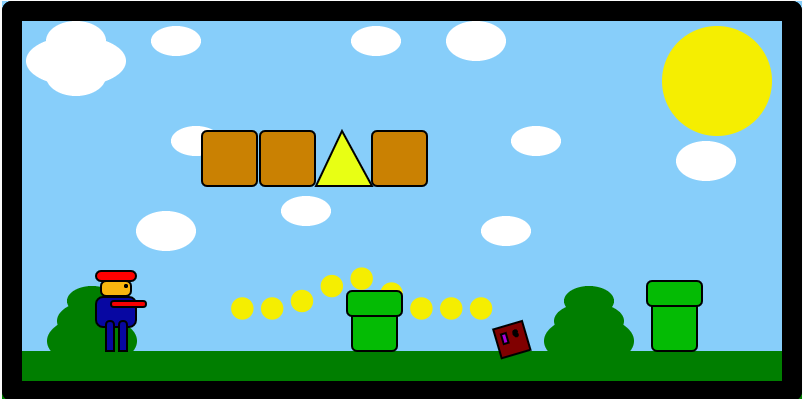
**strokeWeight(2);**

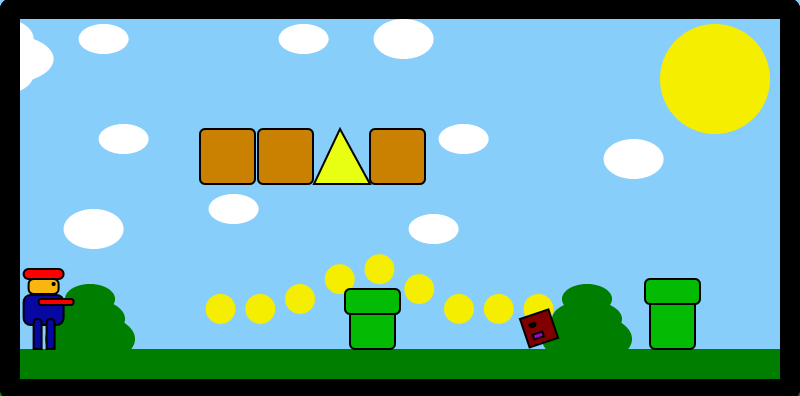
**fill(232,255,20);**

**triangle(314,185,340,130,370,185);**

**}**

**Captura de ecran:**





**Concluzie:**

În concluzie, implementarea transformărilor grafice asupra unei scene 2D cu ajutorul setului de funcționalități oferit de biblioteca p5.js (JavaScript) aduce un nivel semnificativ de dinamică și versatilitate în manipularea obiectelor grafice. Scopul acestui proiect a fost să utilizeze funcționalitățile specifice precum `rotate()`, `scale()`, `translate()`, și `shear()` pentru a efectua transformări grafice 2D în cadrul scenei create la laboratorul nr. 1.

Prin utilizarea acestor funcții, am reușit să obținem o gamă variată de efecte vizuale și animații în cadrul scenei 2D. `rotate()` a fost util pentru implementarea rotațiilor obiectelor în jurul punctelor lor centrale, `scale()` a permis modificarea dimensiunilor acestora, `translate()` a facilitat mutarea oriunde în cadrul scenei, iar `shear()` a oferit posibilitatea de a deforma obiectele într-un mod interesant.

Aceste transformări adaugă o dimensiune suplimentară de creativitate și complexitate în manipularea obiectelor grafice, deschizând uși pentru realizarea unor animații și vizualizări mai captivante. Utilizarea p5.js s-a dovedit a fi un instrument eficient și intuitiv pentru a implementa aceste transformări grafice și a oferi o experiență interactivă și dinamică în cadrul scenei 2D elaborate la laboratorul nr. 1.