<!DOCTYPE html>

<html lang="sv">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Phiculatum Interaktiv</title>

<script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.4.0/p5.js"></script>

<style>

body { text-align: center; font-family: Arial, sans-serif; }

canvas { border: 1px solid #ccc; margin-top: 20px; }

#priceDisplay { margin-top: 10px; font-size: 20px; }

button { margin: 10px; padding: 10px; font-size: 16px; }

select { margin: 10px; padding: 5px; font-size: 16px; }

</style>

</head>

<body>

<h1>Phiculatum - Skapa Din Egen Komposition</h1>

<p>Dra och släpp tavlorna för att skapa din egen komposition.</p>

<div id="priceDisplay">Totalpris: 0 kr</div>

<button onclick="saveComposition()">Spara Komposition</button>

<button onclick="resetComposition()">Återställ</button>

<button onclick="downloadComposition()">Ladda ner som Bild</button>

<select id="compositionList"></select>

<button onclick="loadSelectedComposition()">Ladda Sparad Komposition</button>

<button onclick="deleteSelectedComposition()">Radera Sparad Komposition</button>

<script>

let pieces = [];

let phi = 1.618;

let basePrice = 500;

let totalPrice = 0;

let savedCompositions = [];

function setup() {

createCanvas(800, 500).parent(document.body);

let xStart = 50;

for (let i = 0; i < 6; i++) {

let size = 50 \* pow(phi, i); // Storlek enligt phi-sekvensen

let price = round(basePrice \* pow(phi, i)); // Pris enligt phi

pieces.push(new ArtPiece(xStart, height / 2, size, price));

xStart += size + 20;

}

}

function draw() {

background(240);

for (let piece of pieces) {

piece.show();

}

updatePrice();

}

function mousePressed() {

for (let piece of pieces) {

piece.checkPressed();

}

}

function mouseReleased() {

for (let piece of pieces) {

piece.dragging = false;

}

}

function mouseDragged() {

for (let piece of pieces) {

piece.drag();

}

}

function updatePrice() {

totalPrice = pieces.reduce((sum, piece) => sum + piece.price, 0);

document.getElementById("priceDisplay").innerText = `Totalpris: ${totalPrice} kr`;

}

function saveComposition() {

let composition = pieces.map(piece => ({ x: piece.x, y: piece.y, price: piece.price }));

savedCompositions.push(composition);

updateCompositionList();

alert("Komposition sparad!");

}

function resetComposition() {

let xStart = 50;

for (let i = 0; i < pieces.length; i++) {

pieces[i].x = xStart;

pieces[i].y = height / 2;

xStart += pieces[i].size + 20;

}

updatePrice();

}

function downloadComposition() {

saveCanvas('Phiculatum\_Composition', 'png');

}

function updateCompositionList() {

let select = document.getElementById("compositionList");

select.innerHTML = "";

savedCompositions.forEach((comp, index) => {

let option = document.createElement("option");

option.value = index;

option.text = "Komposition " + (index + 1);

select.appendChild(option);

});

}

function loadSelectedComposition() {

let select = document.getElementById("compositionList");

let index = select.value;

if (index !== "" && savedCompositions.length > 0) {

let selectedComposition = savedCompositions[index];

for (let i = 0; i < pieces.length; i++) {

pieces[i].x = selectedComposition[i].x;

pieces[i].y = selectedComposition[i].y;

}

updatePrice();

alert("Komposition har laddats!");

} else {

alert("Ingen sparad komposition hittades.");

}

}

function deleteSelectedComposition() {

let select = document.getElementById("compositionList");

let index = select.value;

if (index !== "" && savedCompositions.length > 0) {

savedCompositions.splice(index, 1);

updateCompositionList();

alert("Komposition har raderats!");

} else {

alert("Ingen sparad komposition att radera.");

}

}

class ArtPiece {

constructor(x, y, size, price) {

this.x = x;

this.y = y;

this.size = size;

this.price = price;

this.dragging = false;

this.offsetX = 0;

this.offsetY = 0;

}

show() {

fill(0, 200, 100);

rect(this.x, this.y, this.size, this.size);

fill(255);

textSize(14);

textAlign(CENTER, CENTER);

text(`${this.price} kr`, this.x + this.size / 2, this.y + this.size / 2);

}

checkPressed() {

if (mouseX > this.x && mouseX < this.x + this.size &&

mouseY > this.y && mouseY < this.y + this.size) {

this.dragging = true;

this.offsetX = this.x - mouseX;

this.offsetY = this.y - mouseY;

}

}

drag() {

if (this.dragging) {

this.x = mouseX + this.offsetX;

this.y = mouseY + this.offsetY;

}

}

}

</script>

</body>

</html>