PAPER NAME	AUTHOR
RequestFromCourseraPlatform	-
WORD COUNT	CHARACTER COUNT
1985 Words	17651 Characters
PAGE COUNT	FILE SIZE
46 Pages	25.5KB
SUBMISSION DATE	REPORT DATE
Mar 11, 2024 2:17 PM GMT+8	Mar 11, 2024 2:18 PM GMT+8

● 44% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 25% Internet database
- 39% Submitted Works database

● Excluded from Similarity Report

- Bibliographic material
- Quoted material
- Small Matches (Less then 8 words)
- Manually excluded sources

```
/*

T Vickram

The Game Project

ITP 1

*/


var gameChar_x;

var gameChar_y;

var gameChar_width;

var floorPos_y;

var cameraPosX;

var isLeft;

var isRight;

var isFalling;

var isPlummeting;

var inContact;

var collectables;

var canyons;

var clouds;

var mountains;

var trees;

var gameScore;

var flagPole;
var health;

var gameOver;

var sound;

var bgm;

var platforms;

var opponents;

let snowflakes = []



function setup()

{

    createCanvas(1024, 576);

    //inital health

    health = 9

    //inital gameOver false

    gameOver = false

    //initial health variable

    GameStart();


}



function GameStart(){
```

```
floorPos_y = height * 3/4;

gameChar_x = width/2;

gameChar_y = floorPos_y;

gameChar_width = 20;

//initial gameScore

gameScore = 0


isleft = false;

isRight = false;

isFalling = false;

isPlummeting = false;

inContact = false;

cameraPosX = 0;


//objects written below here

collectables = [{x_pos:200,

                 y_pos:floorPos_y-20,

                 size:11,

                 isFound:false,},

                {x_pos:50,

                 y_pos:floorPos_y-20,

                 size:11,

                 isFound:false,},
{x_pos:700,
 y_pos:floorPos_y-20,
 size:11,
 isFound:false,},
{x_pos:1200,
 y_pos:floorPos_y-20,
 size:11,
 isFound:false,},
{x_pos:-400,
 y_pos:floorPos_y-90,
 size:11,
 isFound:false,},
{x_pos:800,
 y_pos:floorPos_y-90,
 size:11,
 isFound:false},
{x_pos:950,
 y_pos:floorPos_y-20,
 size:11,
 isFound:false},
{x_pos:-250,
 y_pos:floorPos_y-20,
 size:11,
            isFound:false},]

canyons = [{x_pos:300,
         width:100},
        {x_pos:800,
         width:100},
        {x_pos: -600,
         width: 300},
        {x_pos: 2000,
         width: 2000},
        {x_pos: -5000,
         width: 3000}]


clouds = [{x_pos:200,
        y_pos:100,
        size:1,},
       {x_pos:800,
        y_pos:200,
        size:1,}]


mountains = [{x_pos:650,
                 y_pos:floorPos_y-138,
                 size:1.5},
                {x_pos:0,
                 y_pos:floorPos_y-138,
                 size:1.5},
                {x_pos:1150,
                 y_pos:floorPos_y-138,
                 size:1.5}]


    trees = [-200,100,410,1200]

    flagPole = {x_pos: -1000,
                isReached: false};

    opponents = [];
    opponents.push(new Opponent(0, floorPos_y - 10, 100));
    opponents.push(new Opponent(-800,floorPos_y-10,100));
    opponents.push(new Opponent(1200,floorPos_y-10,100))

}

function draw()
```

```
{

    ///////////DRAWING CODE//////////
    //code for blue sky
    background(100,155,255);

    //code for the ground
    noStroke();
    //icy blue ground
    fill(162,210,223);
    rect(0, floorPos_y, width, height - floorPos_y);
    //snow covered ground
    fill(255);
    rect(0, floorPos_y, width, height - (floorPos_y+130));


    //scrolling code
    push();
    translate(-cameraPosX,0)


    //draw the canyon
    for(var i=0;i<canyons.length;i++){
```

```
    var i_canyon = canyons[i];

    fill(100,155,255);

    rect(i_canyon.x_pos,floorPos_y,i_canyon.width,height-floorPos_y);


    fill(100,155,255);

    rect(i_canyon.x_pos,floorPos_y+100,i_canyon.width,height-floorPos_y)


    //spike in canyon

    fill(299,0,0);

    noStroke();

    for (var j = 0; j < i_canyon.width / 20; j++) {

        triangle(

            i_canyon.x_pos + j * 20, height,

            i_canyon.x_pos + j * 20 + 10, height - 100,

            i_canyon.x_pos + (j + 1) * 20, height

        );


        //anchor point for canyon

        //fill(255,0,0)

        //ellipse(i_canyon.x_pos,floorPos_y,10,10)

    }

}
```

```
//draw the clouds

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

    var i_cloud = clouds[i]


    fill(255,255,255);

    noStroke();

    ellipse(i_cloud.x_pos * i_cloud.size,

            i_cloud.y_pos * i_cloud.size,

            100*i_cloud.size,

            100*i_cloud.size);

    ellipse(i_cloud.x_pos-40 * i_cloud.size,

            i_cloud.y_pos+0 * i_cloud.size,

            80*i_cloud.size,

            80*i_cloud.size);

    ellipse(i_cloud.x_pos+40 * i_cloud.size,

            i_cloud.y_pos+0 * i_cloud.size,

            80*i_cloud.size,

            80*i_cloud.size);


    i_cloud.x_pos=i_cloud.x_pos+6


    // loop cloud

    if(i_cloud.x_pos > cameraPosX + width){
        i_cloud.x_pos = cameraPosX -20
    }
    //anchor point for clouds
    //fill(255,0,0);
    //ellipse(i_cloud.x_pos, i_cloud.y_pos,10,10)
}

//draw the mountains
for(i=0;i<mountains.length;i++){
    var i_mountain = mountains[i]
    fill(186,242,239)
    triangle(i_mountain.x_pos-150 * i_mountain.size,
             i_mountain.y_pos+92 * i_mountain.size,
             i_mountain.x_pos+50 * i_mountain.size,
             i_mountain.y_pos+92 * i_mountain.size,
             i_mountain.x_pos-50 * i_mountain.size,
             i_mountain.y_pos-110 * i_mountain.size);
    triangle(i_mountain.x_pos-100 * i_mountain.size,
             i_mountain.y_pos+92 * i_mountain.size,
             i_mountain.x_pos+100 * i_mountain.size,
             i_mountain.y_pos+92 * i_mountain.size,
             i_mountain.x_pos+0 * i_mountain.size,
             i_mountain.y_pos-160 * i_mountain.size);
    //white peak of mountain
    fill(255)
    triangle(i_mountain.x_pos-34 * i_mountain.size,
             i_mountain.y_pos-76 * i_mountain.size,
             i_mountain.x_pos-67 * i_mountain.size,
             i_mountain.y_pos-76 * i_mountain.size,
             i_mountain.x_pos-50 * i_mountain.size,
             i_mountain.y_pos-110 * i_mountain.size)
    triangle(i_mountain.x_pos-39 * i_mountain.size,
             i_mountain.y_pos-76 * i_mountain.size,
             i_mountain.x_pos+34 * i_mountain.size,
             i_mountain.y_pos-76 * i_mountain.size,
             i_mountain.x_pos+0 * i_mountain.size,
             i_mountain.y_pos-159 * i_mountain.size)

    //anchor point for mountain
    //fill(255,0,0);
    //ellipse(i_mountain.x_pos-225,floorPos_y,10,10)
}

//draw the trees
for(var i=0;i<trees.length;i++){
    var i_tree = trees[i];

    fill(92, 64, 51);

    rect(i_tree+23,floorPos_y-145+63,40,83);


    fill(58,95,11);

    triangle(i_tree-17,

            floorPos_y-82,

            i_tree+103,

            floorPos_y-82,

            i_tree+43,

            floorPos_y-172)


    triangle(i_tree-17,

            floorPos_y-112,

            i_tree+103,

            floorPos_y-112,

            i_tree+43,

            floorPos_y-202)


}


//draw the collectable

for(var i=0;i<collectables.length;i++){
  var i_collectable = collectables[i]

  if(i_collectable.isFound == false){

    stroke(0);

    fill(225,181,48);

    ellipse(i_collectable.x_pos,

        i_collectable.y_pos,

        3*i_collectable.size);


    stroke(0);

    fill(225,181,48);

    ellipse(i_collectable.x_pos,

        i_collectable.y_pos,

        2*i_collectable.size);


    //anchor point of collectable

    //fill(255,0,0) //ellipse(i_collectable.x_pos,i_collectable.y_pos,10,10)

  }

}


//gameOver text code

if(gameOver){

  DisplayGameOver();

}
```

```
//gameChar anchor point below
//    fill(255,0,0);
//    ellipse(512,432,10,10)



//the game character
if(isLeft && isFalling)
{
    // fox jumping-left code

    //fox head facing left
    //fox nose
    fill(202,78,51)
    rect(gameChar_x-18,gameChar_y-62,3,5)

    //fox face
    fill(250,200,152)
    rect(gameChar_x-15,gameChar_y-64,10,10)

    //fox eyes
    fill(255)
    rect(gameChar_x-15,gameChar_y-64,5,5)
```

```
          fill(0)

          rect(gameChar_x-15,gameChar_y-64,2,2)


          //fox ear

          fill(200,20,54)

          triangle(gameChar_x-15,gameChar_y-65,gameChar_x-11,gameChar_y-

65,gameChar_x-13,gameChar_y-75)

          triangle(gameChar_x-12,gameChar_y-65,gameChar_x-8,gameChar_y-

65,gameChar_x-10,gameChar_y-75)


          //fox body

          fill(200,20,54)

          rect(gameChar_x-5,gameChar_y-66,25,14)


          //fox legs

          /fill(0)

          rect(gameChar_x-3,gameChar_y-53,4,10)

          rect(gameChar_x+2,gameChar_y-53,4,10)

          rect(gameChar_x+11,gameChar_y-53,4,10)

          rect(gameChar_x+16,gameChar_y-53,4,10)


          //jumping lines for fox

          rect(gameChar_x-2,gameChar_y-40,2,25)
```

```
    rect(gameChar_x+8,gameChar_y-40,2,21)

    rect(gameChar_x+18,gameChar_y-40,2,25)


}

else if(isRight && isFalling)

{

    // fox jumping-right code


    //fox nose

    fill(202,78,51)

    rect(gameChar_x+14,gameChar_y-52,3,5)


    //fox face

    fill(250,200,152)

    rect(gameChar_x+4,gameChar_y-54,10,10)


    //fox eyes

    fill(255)

    rect(gameChar_x+9,gameChar_y-54,5,5)

    fill(0)

    rect(gameChar_x+12,gameChar_y-54,2,2)


    //fox ear
```

```
	fill(200,20,54)
	triangle(gameChar_x+8,gameChar_y-55,gameChar_x+12,gameChar_y-
55,gameChar_x+10,gameChar_y-65)
	triangle(gameChar_x+11,gameChar_y-55,gameChar_x+15,gameChar_y-
55,gameChar_x+13,gameChar_y-65)

	//fox body
	fill(200,20,54)
	rect(gameChar_x-20,gameChar_y-56,25,14)

	//fox legs
	/fill(0)
	rect(gameChar_x-18,gameChar_y-43,4,7)

	rect(gameChar_x-13,gameChar_y-43,4,7)
	rect(gameChar_x+-4,gameChar_y-43,4,7)
	rect(gameChar_x+1,gameChar_y-43,4,7)

	//jumping lines for fox
	rect(gameChar_x-17,gameChar_y-30,2,25)
	rect(gameChar_x-7,gameChar_y-30,2,21)
	rect(gameChar_x+3,gameChar_y-30,2,25)
```

```
    }
    9 else if(isLeft)
    {
        // fox walking left code

        //fox nose
        fill(202,78,51)
        rect(gameChar_x-18,gameChar_y-22,3,5)

        //fox face
        fill(250,200,152)
        10 rect(gameChar_x-15,gameChar_y-24,10,10)

        //fox eyes
        fill(255)
        rect(gameChar_x-15,gameChar_y-24,5,5)
        fill(0)
        rect(gameChar_x-15,gameChar_y-24,2,2)

        //fox ear
        fill(200,20,54)
        3 triangle(gameChar_x-15,gameChar_y-25,gameChar_x-11,gameChar_y-
25,gameChar_x-13,gameChar_y-35)
```

```
        triangle(gameChar_x-12,gameChar_y-25,gameChar_x-8,gameChar_y-
25,gameChar_x-10,gameChar_y-35)

        //fox body
        fill(200,20,54)
        rect(gameChar_x-5,gameChar_y-26,25,14)

        //fox legs
        //fill(0)
        rect(gameChar_x-3,gameChar_y-13,4,10)
        rect(gameChar_x+2,gameChar_y-13,4,10)
        rect(gameChar_x+11,gameChar_y-13,4,10)
        rect(gameChar_x+16,gameChar_y-13,4,10)

    }
    else if(isRight)
    {
        // fox walking right code

        //fox nose
        fill(202,78,51)
        rect(gameChar_x+14,gameChar_y-22,3,5)
```

```
          //fox face

          fill(250,200,152)

          rect(gameChar_x+4,gameChar_y-24,10,10)


          //fox eyes

          fill(255)

          rect(gameChar_x+9,gameChar_y-24,5,5)

          fill(0)

          rect(gameChar_x+12,gameChar_y-24,2,2)


          //fox ear

          fill(200,20,54)

          triangle(gameChar_x+8,gameChar_y-25,gameChar_x+12,gameChar_y-
25,gameChar_x+10,gameChar_y-35)

          triangle(gameChar_x+11,gameChar_y-25,gameChar_x+15,gameChar_y-
25,gameChar_x+13,gameChar_y-35)


          //fox body

          fill(200,20,54)

          rect(gameChar_x-20,gameChar_y-26,25,14)


          //fox legs

          /fill(0)
```

```
    rect(gameChar_x-18,gameChar_y-13,4,10)

    rect(gameChar_x-13,gameChar_y-13,4,10)

    rect(gameChar_x+-4,gameChar_y-13,4,10)

    rect(gameChar_x+1,gameChar_y-13,4,10)


}

else if(isFalling || isPlummeting)

{

    // fox jumping facing forwards code


    //body of fox

    fill(200,20,54)

    rect(gameChar_x-6, gameChar_y-63,20,20)


    //face of fox

    fill(250,200,152)

    rect(gameChar_x-4, gameChar_y-60,15,15)


    //left eye of fox

    fill(255)

    rect(gameChar_x-2, gameChar_y-59,5,5)

    fill(0)

    rect(gameChar_x-1, gameChar_y-58,2,2)
```

```
//right eye of fox
fill(255)
rect(gameChar_x+4, gameChar_y-59,5,5)
fill(0)
rect(gameChar_x+5, gameChar_y-58,2,2)

//fox nose
fill(202,78,51)
rect(gameChar_x+1, gameChar_y-54,5,5)

//fox ear
fill(202,78,51)
//left ear
triangle(gameChar_x-2,gameChar_y-60,gameChar_x+6,gameChar_y-
60,gameChar_x+2,gameChar_y-75)
//right ear
triangle(gameChar_x+2,gameChar_y-60,gameChar_x+10,gameChar_y-
60,gameChar_x+6,gameChar_y-75)


//left leg of fox
fill(0)
```

```
    rect(gameChar_x-3,gameChar_y-43,5,7)


    //right leg of fox

    fill(0)

    rect(gameChar_x+6,gameChar_y-43,5,7)


    //jumping lines for fox

    rect(gameChar_x-5,gameChar_y-33,2,25)

    rect(gameChar_x+3,gameChar_y-33,2,21)

    rect(gameChar_x+11,gameChar_y-33,2,25)


}

else

{

    // fox standing front facing code


    //body of fox

    fill(200,20,54)

    rect(gameChar_x-4.7, gameChar_y-28,20,20)


    //face of fox

    fill(250,200,152)

    rect(gameChar_x-3, gameChar_y-25,15,15)
```

```
//left eye of fox
fill(255)
rect(gameChar_x-1, gameChar_y-24,5,5)
fill(0)
rect(gameChar_x, gameChar_y-23,2,2)

//right eye of fox
fill(255)
rect(gameChar_x+5, gameChar_y-24,5,5)
fill(0)
rect(gameChar_x+6, gameChar_y-23,2,2)

//fox nose
fill(202,78,51)
rect(gameChar_x+2, gameChar_y-19,5,5)

//fox ear
fill(202,78,51)
//left ear
triangle(gameChar_x-3,gameChar_y-25,gameChar_x+5,gameChar_y-
25,gameChar_x+1,gameChar_y-38)
//right ear
```

```
        triangle(gameChar_x+5,gameChar_y-25,gameChar_x+13,gameChar_y-
25,gameChar_x+9,gameChar_y-38)


        //left leg of fox
        fill(0)
        rect(gameChar_x-4,gameChar_y-8,5,10)

        //right leg of fox
        fill(0)
        rect(gameChar_x+10,gameChar_y-8,5,10)
    }



    ///////////INTERACTION CODE//////////

    if(isPlummeting){
        gameChar_y +=10;
        isLeft = false;
        isRight = false;

        CheckIfGameCharIsDead();
        return;
```

```
}



//gravity code
if(gameChar_y<floorPos_y){
    inContact = false;
    for(var i=0;i<platforms.length;i++){
        if(platforms[i].checkContact(gameChar_x,gameChar_y) == true){
            inContact = true;
            isFalling = false;
            break;
        }
    }
    if(inContact == false){
        isFalling = true;
        gameChar_y += 2
    }
}
else{
    isFalling = false;
}
//gameChar left right movement code
if(isLeft == true){
    gameChar_x -= 5

    cameraPosX -= 5

}

else if(isRight == true){

    gameChar_x += 5

    cameraPosX += 5

}


//check if game char is in the range of collectable

checkIfGameCharInAnyCollectableRange();


//check if game char is over canyon

checkIfGameCharIsOverCanyons();


//check if game char reached flag pole

CheckIfGameCharReachedFlagPole();


//display game score that doesnt move

DisplayGameScore();


//display flagPole

DisplayFlagPole();
```

```
//gameOver text code
if(gameOver){

    //gameChar respawns to starting point after game over
    10 gameChar_x = width/2;
    gameChar_y = floorPos_y;
    cameraPosX=0;
}

//platform code
5 platforms = [];

platforms.push(createPlatforms(700,floorPos_y-70,350))
platforms.push(createPlatforms(-500,floorPos_y-70,300))

12 for(var i=0;i<platforms.length;i++){
    platforms[i].draw();
}

//opponent code
for(var i=0;i<opponents.length;i++){
    opponents[i].draw();
    var isContact = opponents[i].checkContact(gameChar_x, gameChar_y);


    if(isContact){
        if(health > 0 ){
            GameStart();
            health--
        }
        if(health == 0){
            gameOver = true;
            cameraPosX = 0;
        }
    }
}

pop();



//display game score that follows gameChar
DisplayGameScore();


//display healthbar
DisplayHealthBar();
```

```
    //code for snowflakes

    let t = frameCount/60;


    for(i=0;i<random(5); i++){

        snowflakes.push(new snowflake());

    }


    //loop snowflakes

    for(let flake of snowflakes){

        flake.update(t);

        flake.display();

    }




}




//////function to make gameChar move//////


function keyPressed()

{

    //stop the keys from working when gameOver or isPlummeting

    if(gameOver || isPlummeting == true){
        return;
    }

    // if statements to control the animation of the character when keys are pressed.

    //bgm sound
    backgroundMusic();

    if(keyCode == 37 || keyCode == 65){
        isLeft = true;
    }
    else if(keyCode == 39 || keyCode == 68){
        isRight = true;
    }
    else if(keyCode == 38 || keyCode == 32 || keyCode == 87){
        //ensure char jump when touching grd
        if(gameChar_y>=floorPos_y || inContact == true){
            gameChar_y -= 80;
            //jumping sound
            jumpingSound.play();
        }
    }
}
```

```
function keyReleased()

{

    //stop the keys from working when gameOver

    if(gameOver){

        return;

    }

    if(keyCode == 37 || keyCode == 65){

        //console.log("left arrow");

        isLeft = false;

    }

    else if(keyCode == 39 || keyCode == 68){

        //console.log("right arrow");

        isRight = false;

    }

}



//checking to see if gameChar is over canyons

function checkIfGameCharIsOverCanyons(){

    for(var i=0; i<canyons.length;i++){

        checkIfGameCharIsOverCanyon(canyons[i]);

    }
}


//gameChar plummets if over canyon
function checkIfGameCharIsOverCanyon(i_canyon){
    //check if gameChar x value is more than canyon x value
    var cond1 = gameChar_x-15>i_canyon.x_pos
    //check if gameChar x value is within canyon length
    var cond2 = gameChar_x+20<i_canyon.x_pos + i_canyon.width
    //check if gameChar is on floor
    var cond3 = gameChar_y >= floorPos_y

    //check if game char is over the canyon
    if(cond1 && cond2 && cond3){
        isPlummeting = true
    }

    if(isPlummeting == true){
        fallingSound.play();
    }
}
```

```
////////below is all the functions//////////


//checking to see if GameChar in collectable range
function checkIfGameCharInAnyCollectableRange(){
    for(var i=0;i<collectables.length;i++){
        checkIfGameCharInCollectableRange(collectables[i]);
    }
}


//collect collectable if gameChar is within Range
function checkIfGameCharInCollectableRange(i_collectable){
    if(i_collectable.isFound == false){
        var d = dist(gameChar_x, gameChar_y, i_collectable.x_pos, i_collectable.y_pos)
        if(d<30){
            i_collectable.isFound = true;
            gameScore++;
            collectingSound.play();
        }
    }
}


//shows GameScore
function DisplayGameScore(){
    fill(0);

    textFont(fontNum);

    text(gameScore,115,36)

    textFont(font);

    textSize(30);

    text("Score",10,30);


}


//displays flagpole

function DisplayFlagPole(){

    fill(125,0,0);

6   rect(flagPole.x_pos,floorPos_y-400,20,400);

    noStroke();

    ellipse(flagPole.x_pos+10,floorPos_y-410,40);

    fill(0,128,128);

    if(flagPole.isReached){

        //flag is down

2       rect(flagPole.x_pos,floorPos_y-50,100,50);

    }else{

        //flag is up

        rect(flagPole.x_pos,floorPos_y-390,100,50);

        fill(255)
        textSize(20)

        text("WINNER",flagPole.x_pos+10,floorPos_y-360)

    }

}


//Checking to see if gameChar reached flagPole

function CheckIfGameCharReachedFlagPole(){

    if(flagPole.isReached == false){

        var d = dist(gameChar_x,gameChar_y,flagPole.x_pos,floorPos_y)

        if(d<10){

            flagPole.isReached = true

            //settting GameOver to be true

            gameOver = true;

        }

    }

}


//Checking to see if gameChar is Dead

function CheckIfGameCharIsDead(){

    if(gameChar_y>height){

        //reduce health once gameChar below screen

        if(health > 0){

            health--;
            GameStart();


        }


        if(health == 0){
            gameOver = true;
            cameraPosX = 0;
        }
    }
}


//HealthBar function
function DisplayHealthBar(){
    for(var i=0;i<health;i++){
        fill(139,0,0);
        ellipse(40*i+650,30,20,10);
        fill(100,0,0);
        triangle((40*i+650)+20, 20, (40*i+650)+10, 30, (40*i+650)+20, 40);
    }
}


//gameOver function
function DisplayGameOver(){
```

```
push()

textSize(50)

if(health>0){


    textFont(font);

    textAlign(CENTER);

    stroke(0,255,0);

    strokeWeight(4);


    text("Game Over",width/2,height/2-100);

    text("You WIN",width/2,height/2);

    textFont(fontNum);

    text("Your score is: "+gameScore +"/8",width/2,height/2+50)

}else{

    stroke(255,0,0);

    strokeWeight(4);


    textAlign(CENTER);

    textFont(font);

    text("Game Over",width/2,height/2-100);

    textFont(fontNum);

    text("You LOST all your 9 lives",width/2,height/2)

}
```

```
    pop()
}



//sound and music function
function backgroundMusic(){
    if(!bgm.isPlaying()){
        bgm.play();
        bgm.loop();
        bgm.setVolume(0.2)
    }



}

//preload function for sound and music
function preload(){
    //bgm is from youtube
    bgm = loadSound("music/bgm.mp3");

    //jumping sound from youtube royalty free collection
    jumpingSound = loadSound("music/jumping.mp3");
    jumpingSound.setVolume(0.3);
    //collecting sound from mixkit.co royalty free selection

    collectingSound = loadSound("music/collectable.mp3");

    collectingSound.setVolume(0.3);


    //falling sound form mixkit.co royalty free selection

    fallingSound = loadSound("music/Falling.mp3");

    fallingSound.setVolume(0.3);


    font = loadFont("font.ttf");

    fontNum = loadFont("fontNum.ttf")

}



//platform creation function

function createPlatforms(x, y, length){

    var p = {

        x: x,

        y: y,

        length: length,

        draw: function(){

            fill(83,204,220);

            rect(this.x,this.y,this.length,20);
            fill(255);
            rect(this.x,this.y,this.length,5)

        },
        checkContact: function(gameChar_X,gameChar_Y){
            if(gameChar_X>this.x && gameChar_X<this.x + this.length){
                var d = this.y - gameChar_Y;
                if(d>=0 && d<5){
                    return true;
                }
            }
            return false;
        }

    }
    return p;
}



//function code for opponent
function Opponent(x, y, range){
    this.x = x;
    this.y = y;
    this.range = range;


    this.currentX = x;

    this.inc = 1;


    this.update = function(){

        this.currentX += this.inc;


        if(this.currentX >= this.x + this.range){

            this.inc = -1

        }

        else if(this.currentX < this.x){

            this.inc = 1;

        }

    }


    this.draw = function(){

        this.update();

        opponentFox(this.currentX, this.y);


    }


    this.checkContact = function(gameChar_x, gameChar_y){
        var d = dist(gameChar_x, gameChar_y, this.currentX, this.y)

        if(d<20){
          return true;
        }
        return false;
      }
    }



    //function code to draw opponenet
    function opponentFox(x, y) {
      // fox standing front facing code

      //body of fox
      fill(200, 20, 54)
      rect(x - 4.7, y - 28, 20, 20)

      //face of fox
      fill(250, 200, 152)
      rect(x - 3, y - 25, 15, 15)

      //left eye of fox
```

```
fill(255)

rect(x - 1, y - 24, 5, 5)

fill(0)

rect(x, y - 23, 2, 2)


//right eye of fox

fill(255)

rect(x + 5, y - 24, 5, 5)

fill(0)

rect(x + 6, y - 23, 2, 2)


//fox nose

fill(202, 78, 51)

rect(x + 2, y - 19, 5, 5)


//fox ear

fill(202, 78, 51)

//left ear

triangle(x - 3, y - 25, x + 5, y - 25, x + 1, y - 38)

//right ear

triangle(x + 5, y - 25, x + 13, y - 25, x + 9, y - 38)


//left leg of fox
```

```
    fill(0)

    rect(x - 4, y - 8, 5, 15)


    //right leg of fox

    fill(0)

    rect(x + 10, y - 8, 5, 15)

}




//function code for snowflake

function snowflake()

{

    fill(255);

    // initialize coordinates

    this.posX = 0;

    this.posY = random(-50, 0);

    this.initialangle = random(0, 2 * PI);

    this.size = random(2, 5);


    // snowflake radius

    // chosen so the snowflakes are uniformly spread out in area

    this.radius = sqrt(random(pow(width / 2, 2)));
    this.update = function(time) {
        // x position follows a circle
        let w = 0.6; // angular speed
        let angle = w * time + this.initialangle;
        this.posX = width / 2 + this.radius * sin(angle);

        // different size snowflakes fall at slightly different y speeds
        this.posY += pow(this.size, 0.5);

        // delete snowflake if past end of screen
        if (this.posY > floorPos_y)
        {
            let index = snowflakes.indexOf(this);
            snowflakes.splice(index, 1);
        }
    };

    this.display = function() {
        ellipse(this.posX, this.posY, this.size);
    };
}
```

44% Overall Similarity

Top sources found in the following databases:

- 25% Internet database
- 39% Submitted Works database

TOP SOURCES

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	**courses.ideate.cmu.edu** Internet	5%
2	**uol-bcs on 2023-04-02** Submitted works	4%
3	**uol-bcs on 2023-07-06** Submitted works	3%
4	**uol-bcs on 2024-01-06** Submitted works	2%
5	**uol-bcs on 2023-03-07** Submitted works	2%
6	**uol-bcs on 2023-07-15** Submitted works	2%
7	**uol-bcs on 2021-09-13** Submitted works	2%
8	**uol-bcs on 2022-12-31** Submitted works	2%
9	**uol-bcs on 2023-07-05** Submitted works	2%

10	uol-bcs on 2023-09-18 Submitted works	1%
11	uol-bcs on 2020-01-10 Submitted works	1%
12	uol-bcs on 2023-03-26 Submitted works	1%
13	uol-bcs on 2024-01-08 Submitted works	1%
14	uol-bcs on 2024-01-02 Submitted works	1%
15	uol-bcs on 2022-12-15 Submitted works	1%
16	uol-bcs on 2022-12-25 Submitted works	1%
17	uol-bcs on 2023-03-27 Submitted works	1%
18	uol-bcs on 2023-07-03 Submitted works	1%
19	uol-bcs on 2023-01-09 Submitted works	<1%
20	uol-bcs on 2024-01-06 Submitted works	<1%
21	uol-bcs on 2021-03-28 Submitted works	<1%

22	uol-bcs on 2022-03-28 Submitted works	<1%
23	uol-bcs on 2022-12-18 Submitted works	<1%
24	uol-bcs on 2023-01-03 Submitted works	<1%
25	uol-bcs on 2023-07-12 Submitted works	<1%
26	uol-bcs on 2023-01-08 Submitted works	<1%
27	uol-bcs on 2023-03-25 Submitted works	<1%
28	uol-bcs on 2023-03-11 Submitted works	<1%
29	uol-bcs on 2023-03-12 Submitted works	<1%
30	uol-bcs on 2023-12-21 Submitted works	<1%
31	uol-bcs on 2024-01-06 Submitted works	<1%
32	gitlab.doc.gold.ac.uk Internet	<1%
33	uol-bcs on 2021-09-10 Submitted works	<1%

34	uol-bcs on 2023-07-16 Submitted works	<1%
35	uol-bcs on 2024-01-03 Submitted works	<1%

● Excluded from Similarity Report

- Bibliographic material
- Small Matches (Less then 8 words)
- Quoted material
- Manually excluded sources

EXCLUDED OVERLAPPING SOURCES

Submitted to uol-bcs on 2024-03-11 Submitted works	81%
Submitted to uol-bcs on 2024-01-08 Submitted works	50%
Submitted to uol-bcs on 2024-01-06 Submitted works	49%