//currently the players can 1) not move 2) move on top of each other 3) sit forever trying to figure out a move

let size = 50;

let width = 70;

let height = 70;

let numNodes = 19;

let tbl = [[0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0],

[1,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0],

[0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0],

[1,0,0,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0],

[1,1,0,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0],

[0,1,1,0,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0],

[0,0,1,0,0,1,0,0,0,0,1,1,0,0,0,0,0,0,0],

[0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0],

[0,0,0,1,1,0,0,1,0,1,0,0,1,1,0,0,0,0,0],

[0,0,0,0,1,1,0,0,1,0,1,0,0,1,1,0,0,0,0],

[0,0,0,0,0,1,1,0,0,1,0,1,0,0,1,1,0,0,0],

[0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0],

[0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0],

[0,0,0,0,0,0,0,0,1,1,0,0,1,0,1,0,1,1,0],

[0,0,0,0,0,0,0,0,0,1,1,0,0,1,0,1,0,1,1],

[0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,0,0,0,1],

[0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,1,0],

[0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,1],

[0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0]]

let bd;

let activeBlockers = new Array();

let ply1, ply2;

let gameOver = false;

let playerTurn = 1;

function setup(){

createCanvas(500,500);

background(255);

bd = new Board();

ply1 = new Player(1);

ply2 = new Player(2);

ply1.turn1();

ply2.turn1();

playerTurn = 0;

}

function draw(){

if (playerTurn == 1){

ply1.play();

playerTurn = 2;

} else{

ply2.play();

playerTurn = 1;

}

if(gameOver == true){

noLoop();

return;

}

bd.show();

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

stroke(0);

strokeWeight(.5);

fill(0);

text(i, bd.platforms[i].x - 5,bd.platforms[i].y-10);

}

for(var j = 0; j < activeBlockers.length; j++){

activeBlockers[j].show();

}

ply2.show();

ply1.show();

}

function isPath(current, desired){

if(current == desired){

return true;

}

var queue = new Queue();

var visited = new Array(numNodes);

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

visited[i] = false;

}

visited[current] = true;

queue.enqueue(current);

while(queue.qList.length > 0){

current = queue.qList[queue.head];

queue.dequeue();

for(var i = 0; i < tbl[current].length; i++){

if(tbl[current][i] == 1 && ply1.plat.id != i && ply2.plat.id != i){

if(i == desired){

return true;

}

if(!visited[i]){

visited[i] = true;

queue.enqueue(i);

}

}

}

}

return false;

}

class Board{

constructor(){

this.platforms = new Array();

for(var i = 0; i < numNodes; i++){

if(i<3){

this.platforms.push( new platform(width\*(i+1),height,i)); //3 row

}

else if(i < 7){

this.platforms.push( new platform(width\*(i-3)+(width/2), 2\*height,i)); //4 row

}

else if(i < 12){

this.platforms.push( new platform(width\*(i-7), 3\*height,i)); //5 row

}

else if(i < 16){

this.platforms.push( new platform(width\*(i-12)+(width/2),4\*height,i)); //4 row

}

else{

this.platforms.push( new platform(width\*(i-15),5\*height,i)); //3 row

}

}

}

show(){

strokeWeight(3);

stroke(255,102,0);

for(var i = 0; i < numNodes; i++){

this.platforms[i].show();

for(var j = 0; j< numNodes; j++){

if (tbl[i][j] == 1){

line(this.platforms[i].x, this.platforms[i].y, this.platforms[j].x, this.platforms[j].y);

}

}

}

}

}

class platform{

constructor(x,y,id){

this.x = x + 100;

this.y = y;

this.id = id;

}

show(){

fill(255,102,0);

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

}

}

class Blocker{

constructor(a,b){

this.between = [a.id,b.id];

if(tbl[a.id][b.id] == 1 || tbl[b.id][a.id] == 1){

tbl[a.id][b.id] = 0;

tbl[b.id][a.id] = 0;

this.x = (a.x + b.x)/2;

this.y = (a.y + b.y)/2;

activeBlockers.push(this);

this.show();

}

}

show(){

noStroke();

fill(150,145,145);

ellipse(this.x, this.y, size/4);

}

}

class Player{

constructor(n){

this.id = n;

this.moves = new Array();

this.blockers = new Array();

if (n == 1){

this.plat = bd.platforms[7];

}

else{

this.plat = bd.platforms[11];

}

}

show(){

fill(0);

rect(this.plat.x-5, this.plat.y-5, 10,10);

}

turn1(){

this.addBlocker(4);

}

play(){

var beginningPed = this.plat;

var possibleMoves = new Array();

for(var i = 0; i < numNodes; i++){

if(i != ply1.plat.id && i != ply2.plat.id && isPath(this.plat.id,i)){

possibleMoves.push(i);

}

}

if(possibleMoves.length == 0){

console.log("GAME OVER");

gameOver = true;

return;

}

var a = random(possibleMoves);

this.plat = bd.platforms[a];

console.log("Player "+ this.id + " moved from " + beginningPed.id + " to " + a);

this.moves.push(a);

this.addBlocker(1);

}

addBlocker(n){

var blockerCounter = 0;

while(blockerCounter < n){

var a = Math.floor(Math.random()\*numNodes);

var b = Math.floor(Math.random()\*numNodes);

if(bd.platforms[a].id != 7 && bd.platforms[b].id != 7 && bd.platforms[a].id != 11 && bd.platforms[b].id != 11 && tbl[a][b] == 1)

{

console.log("Player " + this.id + " placed blocker between " + bd.platforms[a].id + " and "+ bd.platforms[b].id);

var x = new Blocker(bd.platforms[a], bd.platforms[b]);

this.blockers.push(x);

blockerCounter++;

}

}

}

}