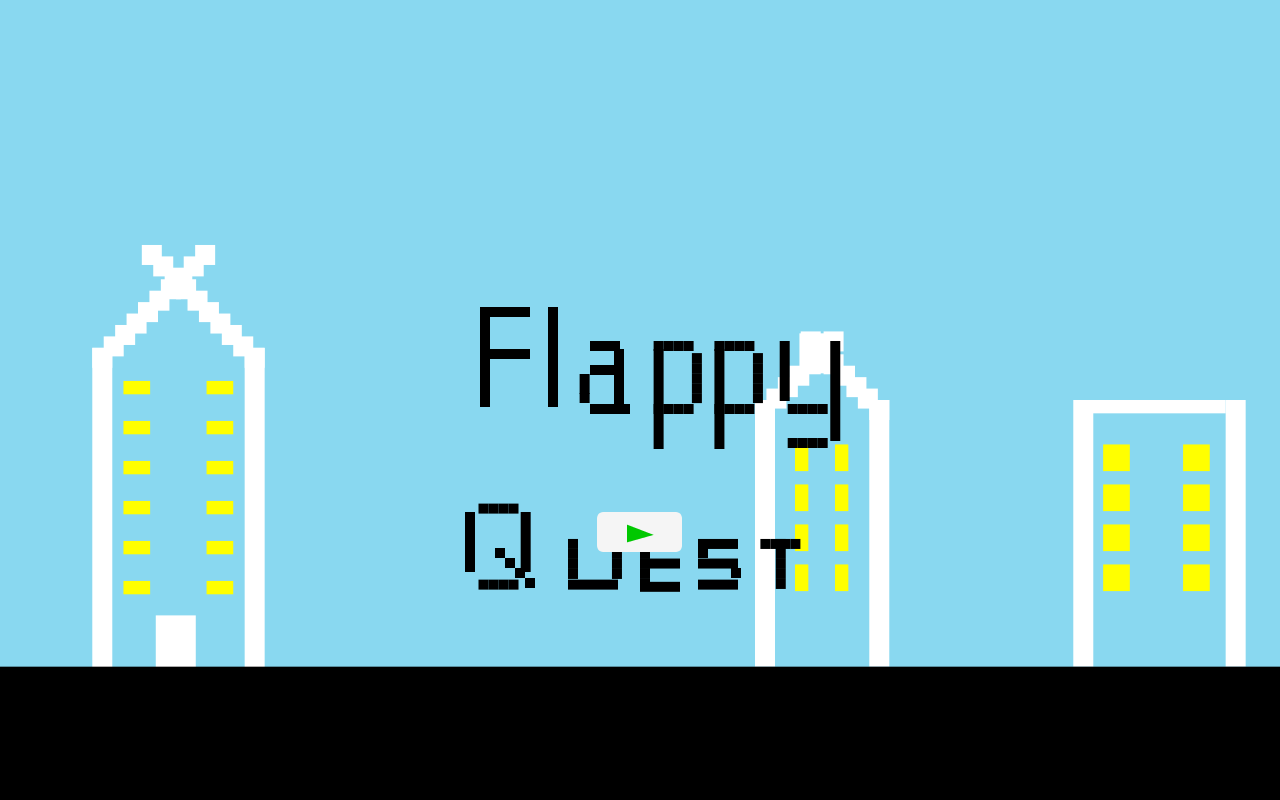
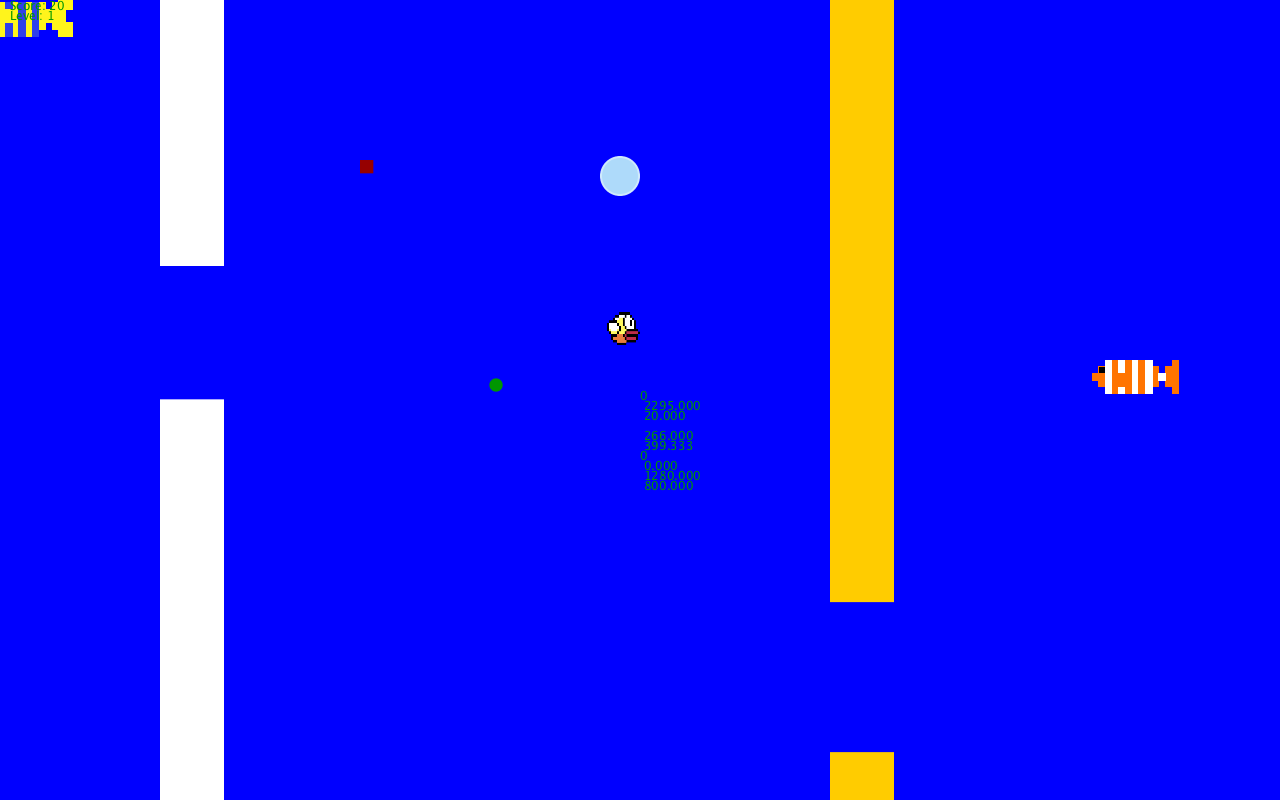
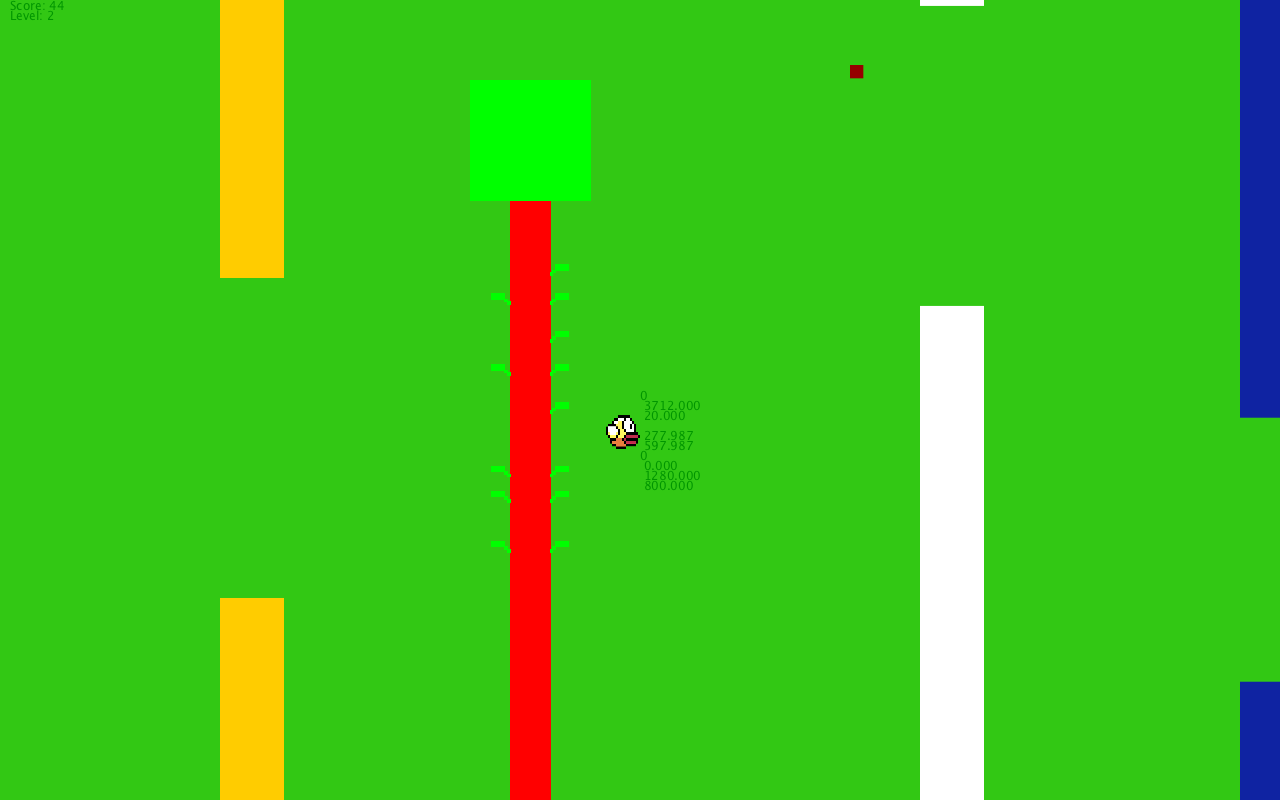
****

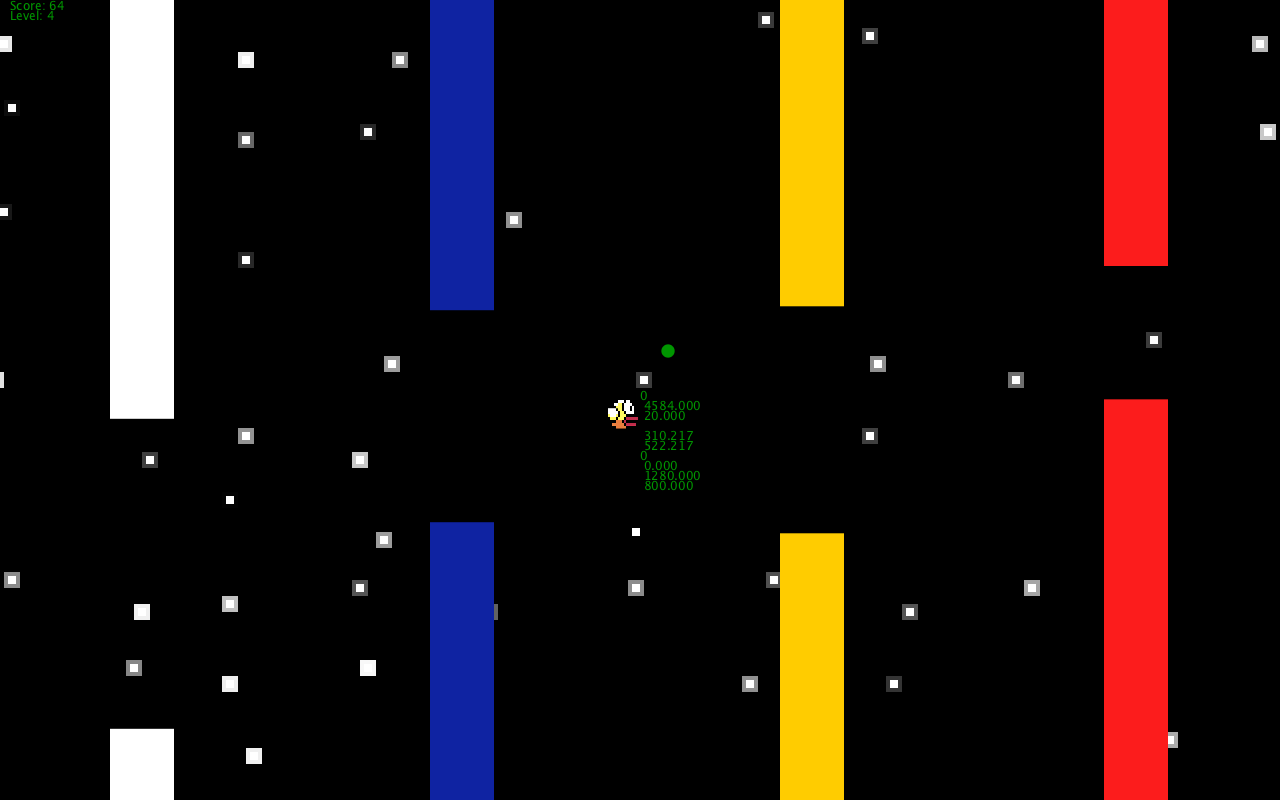
Group member names and IDs:

* A1669696 - [Ashley Jayasuriya](https://www.facebook.com/100002706567939)
* A1714190 - Nalita Sok
* A1711891 - [Diana Marcela Guevara](https://www.facebook.com/791664198)

****

****

****

****

****

**DESCRIPTION OF DESIGN -FLAPPY QUEST**

(decomposition)

Start Menu

State = 0

Click on PLAY bottom

Menu

State = 0.

Columns and background.

State = 1.

Click on flappy bird avoiding the columns

Game Over Menu

State = 0.

collisioncheckmouseclickball1==0

collisioncheckmouseclickcolumn3=0

collisioncheckmouseclickball1==1

collisioncheckmouseclickcolumn3=1

Yes

No

No

Yes

If score increase

No

Yes

Change the background until level is == 3. After keep the last background

If level increase == to a fixed score

**TESTING PROCESS**

**We make the following test cases:**

1. Making sure that the ball is moving properly when we press the mouse.
2. Test to see if the column properly generate randomly.
3. Check if the animated background is moving.

-There are 4 levels in the game:

+ The city level

+ Underwater level

+ Forest level

+ Space level

4. We change the following code to test the game after ensemble

class Birb {

(…)

void update() {

(…)

yPos1 = mouseY;//yPos1+valy;

5. The level of difficulty increases as the level increase. We increment a new column every single time the level increases until it reaches level 3.

6. They are 2 black pills. One reduces the size of the bird for a couple of seconds, the other triggers the ground to turn red and if the bird touches the ground the game will end.

7. We tried to test the game many times in order to fix every bug and error that was found.

**HOW FLAPPY QUEST WAS GENERATED?**

**I. PLANNING**

**1. Main Variables:**

|  |
| --- |
| int checkmouseclick=0;//mouse click for jump up |
| int checkmouseclick2=0;//ensures jump lasts a certain number of frames |
| float speedcounter;//counts frames for speed calculation |
| float speed;//frames divided by 6 for speed |
| float collisioncheckmouseclickball1=0;//checks if columns are close enough to hit ball |
| float collisioncheckmouseclickcolumn1;//checks end of top 'column' |
| float collisioncheckmouseclickcolumn2;//checks end of bottom 'column' |
| int collisioncheckmouseclickcolumn3=1;//checks whether ball is safely between 'columns' |
| int score;//starting score |
| int level; |
| int state=0;//game menu/other states |
| Columns c1; |
| Columns c2; |
| Columns c3; |
| Columns c4; |
| Birb b1; |

**2. Main Classes:**

**class Columns {**

float valx, tru, xPos, yPos, heigh, gap, lim; color basic;

Columns (float xStart, float yStart, float high, color base) {

valx=-5;

xPos = xStart;

yPos = yStart;

heigh=high;

basic=base;

gap=width/6;}

void update() {

xPos = xPos+valx;

noStroke();

fill(basic);

rect(xPos, yPos, width/20, heigh);

rect(xPos, heigh+gap, width/20, width-heigh-gap);

if (xPos==-(width/20)) {

xPos=width+20;

heigh=random(0,height\*(5/6)+0.001);

if (heigh<=7\*width/12) {

lim=251;}

else {

lim=width-heigh; }

gap=random(100,lim);}

if (xPos<=width/2+25 && xPos>=width/2-30) {

collisioncheckmouseclickball1=1;

collisioncheckmouseclickcolumn1=heigh;

collisioncheckmouseclickcolumn2=heigh+gap;

if (b1.yPos1-25>collisioncheckmouseclickcolumn1 && b1.yPos1+25<collisioncheckmouseclickcolumn2) {

collisioncheckmouseclickcolumn3=1} else {

collisioncheckmouseclickcolumn3=0;}

**2. class Birb** {

float xPos1, yPos1, valy, valx;

Birb (float xStart, float yStart) {

valx=0;

valy=speed;

xPos1 = xStart;

yPos1 = yStart; }

void update() {

if (checkmouseclick==1 && checkmouseclick2!=5) {

speedcounter=-33;

checkmouseclick2++;

if (checkmouseclick2==5) {

checkmouseclick=0;

checkmouseclick2=0;

valy=speed;}

speedcounter=speedcounter+3;

speed=speedcounter/6;

if (speed>20) {

speed=20.0;}

valy=speed;

xPos1 = xPos1+valx;

yPos1 = yPos1+valy;

if (yPos1>575) {

yPos1=575;}

if (yPos1<25) {

yPos1=25;}

fill(255, 0, 0);

ellipse(xPos1, yPos1, 50, 50);

**II. SETUP AND DRAW**

**1. void setup()** {

fullScreen();

background(255);

frameRate(30);

img = loadImage("flappybird\_topic.png");

if (width>height) {

ssq=height;}

else {

ssq=width}

score=0;

x=width;

y=height;

c1 = new Columns(x, 0, height/3, #FFCC00);

c2 = new Columns(x, 0, height/3, #FFFFFF);

c3 = new Columns(1.25\*x, 0, height/3, #0F23A2);

c4 = new Columns(2.3\*x, 0, height/3, #FC1C1C);

b1 = new Birb(x/2, y/2);

bld1= new Buildingone(x\*0.04, 1); //xpos

bld2= new Buildingtwo(x\*0.2, 1); //xpostwo

bld3= new Buildingthree(x\*0.6, 1); //xposthree

back=#82C2F2;

bubble1=#CFE9FC;

bubble2=#AEDAFA;

bub1 = new BgUWBub(width/2,height/2, 20);

f1 = new BgUWFish(100,100,0);

f2 = new BgUWFish(400,400,1);

f3 = new BgUWFish(300,300,2);

ts1 = new TwinklingStars(width/10\*0);

ts2 = new TwinklingStars(width/10\*1);

ts3 = new TwinklingStars(width/10\*2);

ts4 = new TwinklingStars(width/10\*3);

ts5 = new TwinklingStars(width/10\*4);

ts6 = new TwinklingStars(width/10\*5);

ts7 = new TwinklingStars(width/10\*6);

ts8 = new TwinklingStars(width/10\*7);

ts9 = new TwinklingStars(width/10\*8);

ts10 = new TwinklingStars(width/10\*9);

ts11 = new TwinklingStars(width);

t1=new trees(width\*0.9,10);

t2=new trees(width\*0.7,10);

t3=new trees(width\*0.5,10);

t4=new trees(width\*0.3,10);

counter=0;

counter1=0;

limit=10;

limita=10;

limitb=10;

size=height/24;}

**2. void draw() {**

if (state==0) {

fill(#89d8f0);

rect(0, 0, width, height);

rect(ssq/3, height/6, ssq/3, height\*2/3);

height/20,7);

bld1.update();

bld2.update();

bld3.update();

ground();

fill(0);

for(int i=0;i<10;i++){

}

//start menu

else {

if (collisioncheckmouseclickball1==1) {

if (collisioncheckmouseclickcolumn3==1) {

if (level<1) {

fill(#000080);

rect(0, 0, width, height);

bld1.update();

bld2.update();

bld3.update();

ground();

c1.update();

}

else if (level==1) {

fill(0,0,255);

rect(0, 0, width, height);

bub1.update();

f1.update();

f2.update();

f3.update();

c1.update();

c2.update();

}

else if (level==2) {

fill(50,200,20);

rect(0,0,width,height);

t1.update();

t2.update();

t3.update();

t4.update();

c1.update();

c2.update();

c3.update();

}

else if (level>2) {

fill(0);

rect(0,0,width,height);

ts1.update();

ts2.update();

ts3.update();

ts4.update();

ts5.update();

ts6.update();

ts7.update();

ts8.update();

ts9.update();

ts10.update();

ts11.update();

c1.update();

c2.update();

c3.update();

c4.update(); }

b1.update();

red\_pill();

green\_pill();

text(checkmouseclick, width/2, height/2);

text(speedcounter, width/2, height/2+10);

text(speed, width/2, height/2+20);

//text(valy,width/2,height/2+30);

text(collisioncheckmouseclickcolumn1, width/2, height/2+40);

text(collisioncheckmouseclickcolumn2, width/2, height/2+50);

text(collisioncheckmouseclickcolumn3, width/2, height/2+60);

text(collisioncheckmouseclickball1, width/2, height/2+70);

text(x, width/2, height/2+80);

text(y, width/2, height/2+90);

score=score+1;

level=floor(score/16);

text("Score: "+score, 10, 10);

text("Level: "+level, 10, 20);

collisioncheckmouseclickball1=0;

collisioncheckmouseclickcolumn3=0;

}//what to do if ball passes between columns

else {

for (int i=0; i<11; i++) {

for (int j=0; j<test[i].length; j++) {

if (test[i][j]==0) {

fill(200);

rect((width/2)-100+(10\*j),(height/2)-100+(10\*i),10,10);}

state=2;

}//what to do if the ball does not pass between columns}

}//what to do if the ball does not pass between columns

} else {

if (level<1) {

fill(#000080);

rect(0, 0, width, height);

bld1.update();

bld2.update();

bld3.update();

ground();

c1.update();}

else if (level==1) {

fill(0,0,255);

rect(0, 0, width, height);

bub1.update();

f1.update();

f2.update();

f3.update();

c1.update();

c2.update(); }

else if (level==2) {

fill(50,200,20);

rect(0,0,width,height);

t1.update();

t2.update();

t3.update();

t4.update();

c1.update();

c2.update();

c3.update();}

else if (level>2) {

fill(0);

rect(0,0,width,height);

ts1.update();

ts2.update();

ts3.update();

ts4.update();

ts5.update();

ts6.update();

ts7.update();

ts8.update();

ts9.update();

ts10.update();

ts11.update();

c1.update();

c2.update();

c3.update();

c4.update();}

b1.update();

red\_pill();

green\_pill();

text(checkmouseclick, width/2, height/2);

text(speedcounter, width/2, height/2+10);

text(speed, width/2, height/2+20);

//text(valy,width/2,height/2+30);

text(collisioncheckmouseclickcolumn1, width/2, height/2+40);

text(collisioncheckmouseclickcolumn2, width/2, height/2+50);

text(collisioncheckmouseclickcolumn3, width/2, height/2+60);

text(collisioncheckmouseclickball1, width/2, height/2+70);

text(x, width/2, height/2+80);

text(y, width/2, height/2+90);

text("Score: "+score, 10, 10);

text("Level: "+level, 10, 20);

}//what to do while ball is not close to columns

}//gameplay state

}

**3. void mouseClicked() {**

if (state==0) {

if (mouseX<width\*8/15 && mouseX>width\*7/15 && mouseY<height\*69/100 &&mouseY>height\*16/25) {

state=1;

score=0;

gree\_pill\_y = int(random(50,height-50));

gree\_pill\_x = width;

red\_pill\_y = int(random(50,height-50));

red\_pill\_x = width+200;

size = height/24;

}

}//start game

else if (state==1) {

checkmouseclick=1;

speedcounter=0;

}//click event which pushes ball up

else {

if (mouseX<width\*3/4 && mouseX>width/4 && mouseY<height\*3/4 && mouseY>height/4) {

state=0;

gree\_pill\_y = int(random(50,height-50));

gree\_pill\_x = width;

red\_pill\_y = int(random(50,height-50));

red\_pill\_x = width+200;

size = height/24;

float[] randomiseheight = new float[5];

float[] randomisegap = new float[5];

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

float lim=0;

randomiseheight[i]=random(0,500.001);

if (randomiseheight[i]<350) {

lim=251;

}

else {

lim=width-randomiseheight[i];

}

randomisegap[i]=random(100,lim);

}

b1.yPos1=x/2;

c1.xPos=x;

c1.heigh=randomiseheight[0];

c1.gap=randomisegap[0];

c2.xPos=x;

c2.heigh=randomiseheight[1];

c2.gap=randomisegap[1];

c3.xPos=x\*13/10;

c3.heigh=randomiseheight[2];

c3.gap=randomisegap[2];

c4.xPos=x\*17/10;

c4.heigh=randomiseheight[3];

c4.gap=randomisegap[3];

collisioncheckmouseclickcolumn3=0;

collisioncheckmouseclickball1=0;

score=0;

level=0;

}

}//restart game after loss

}