Teaching Code

Arthur Carabott

April 2015

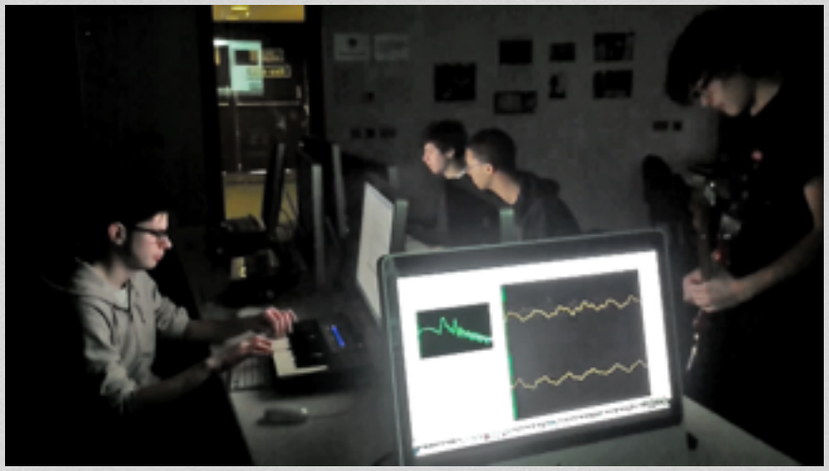
@acarabott
arthurcarabott.com

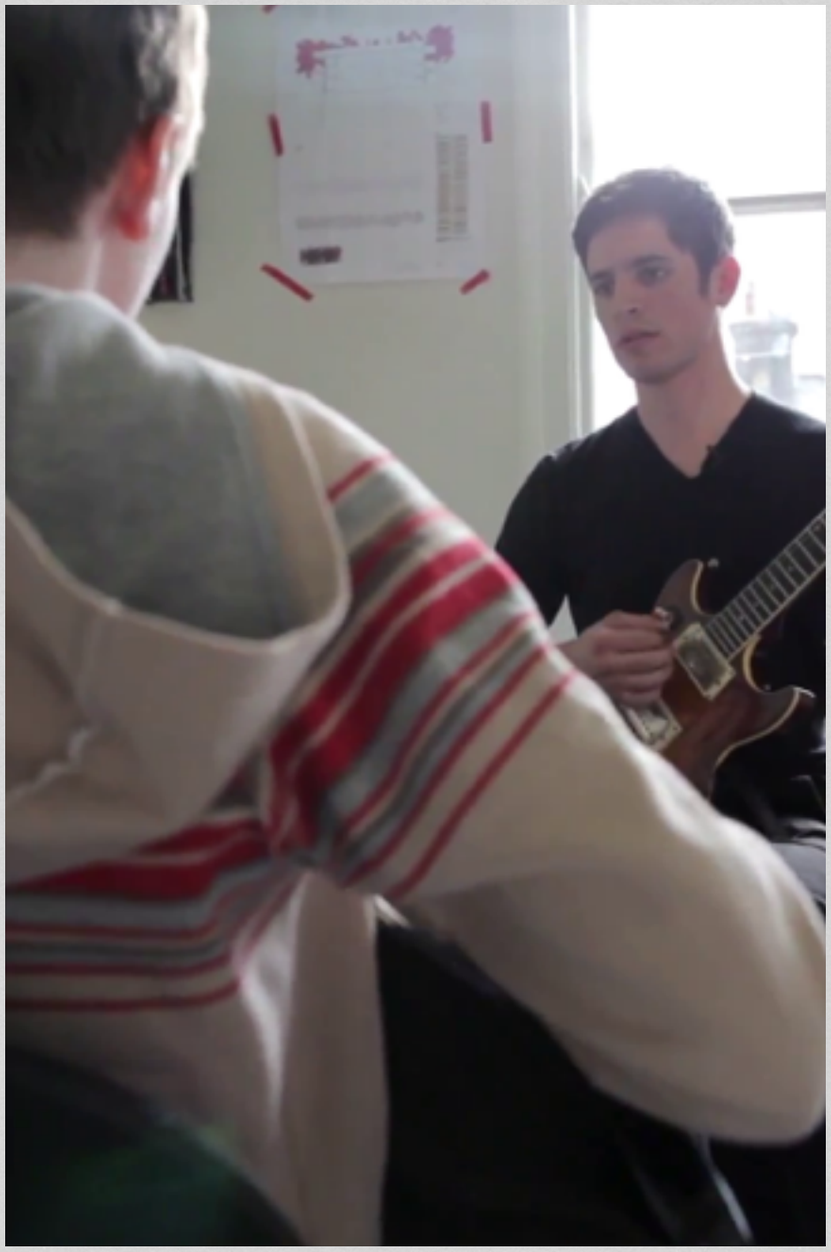
My Teaching

Code

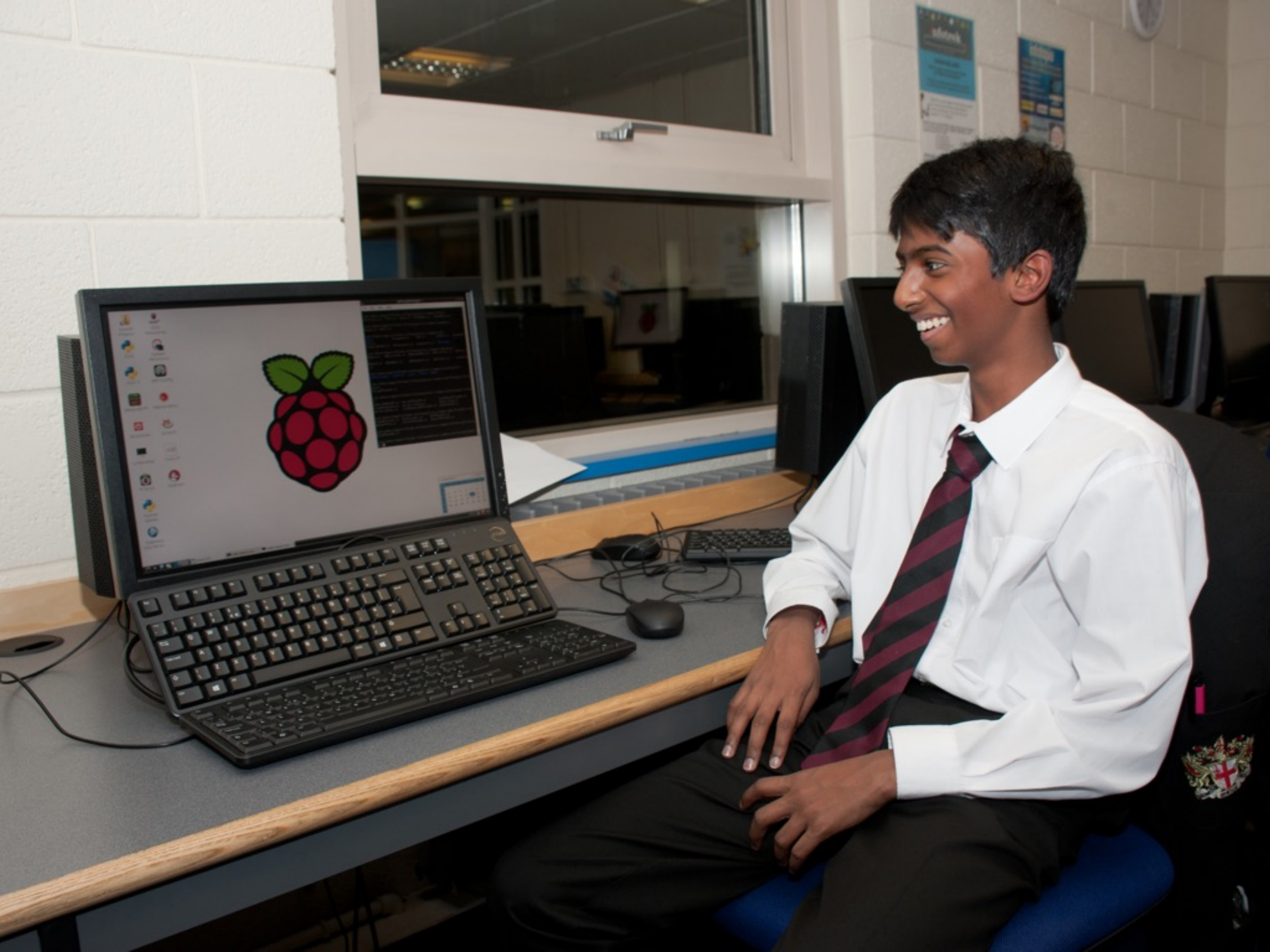
Code+Music

Music





Resonate

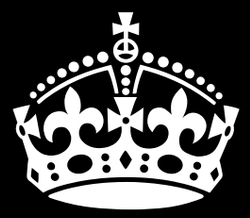


- Why teach?
- What to teach
- Am I wrong?

Why teach?

The whole art of teaching
is only the art of
awakening the natural
curiosity of young minds
for the purpose of
satisfying it afterwards.

-Anatole France



KEEP

CALM

I'M

ONLY

KIDDING

...ish

LASER RIFT

PLAY

EXIT

Things that you will be called out on:

Your knowledge

Your assumptions

Things you dont have to do when coding alone:

Explain yourself
Show your code
Admit what you dont know

Kids get "Creative Coding"

```java
import java.util.ArrayList;

/**
 * Class DrinksMachine simulates a vending machine selling cans of coke.
 *
 */

public class DrinksMachine
{

    private CanOfCoke[] cans;
    private ArrayList<Coin> cashBox;
    private int costOfCan;
    // balance stores value of coins inserted before a can is bought
    private int balance;
    // next points at next can of coke to be sold
    private int next;
```

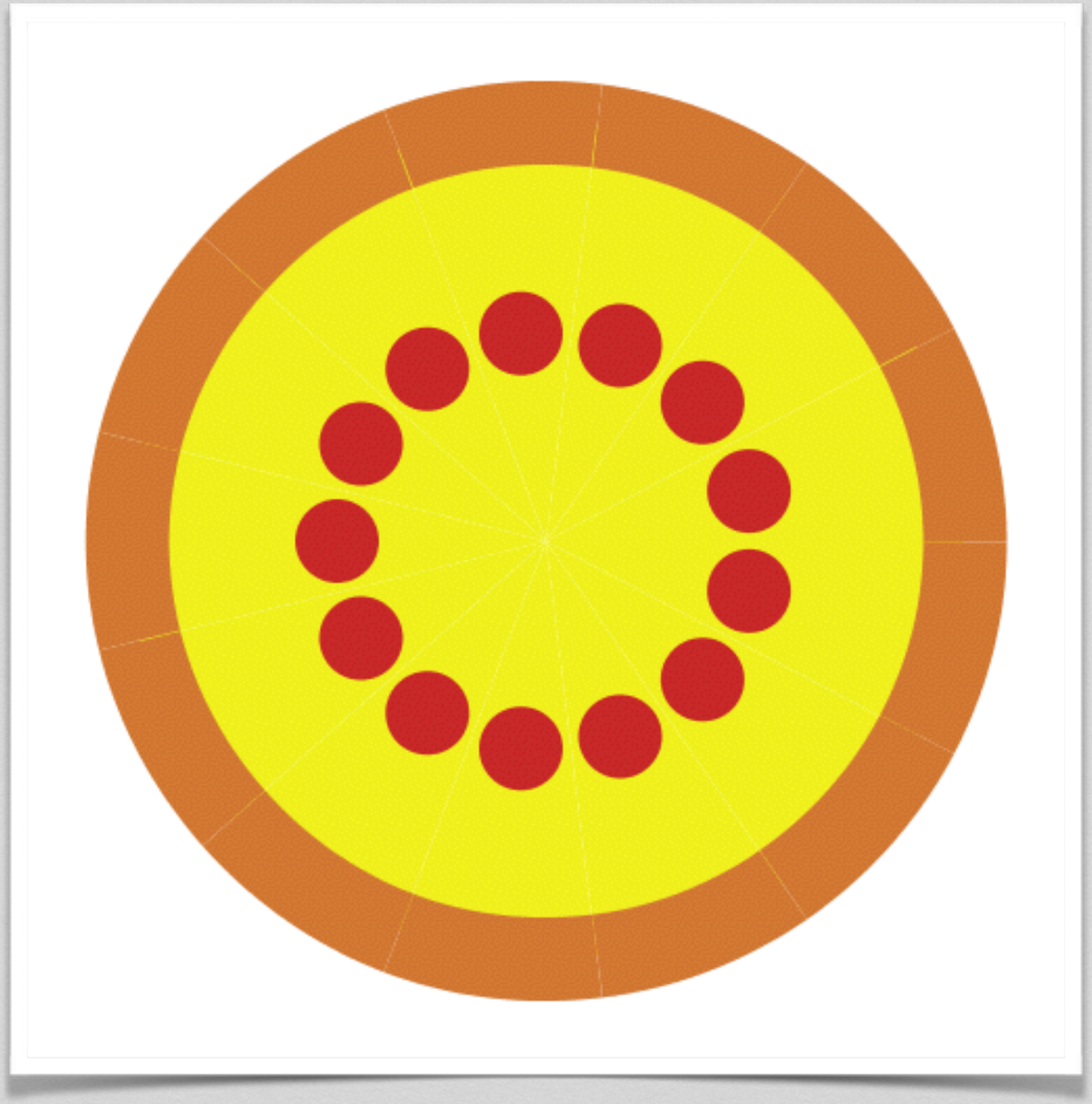
Decode by Karsten Schmidt

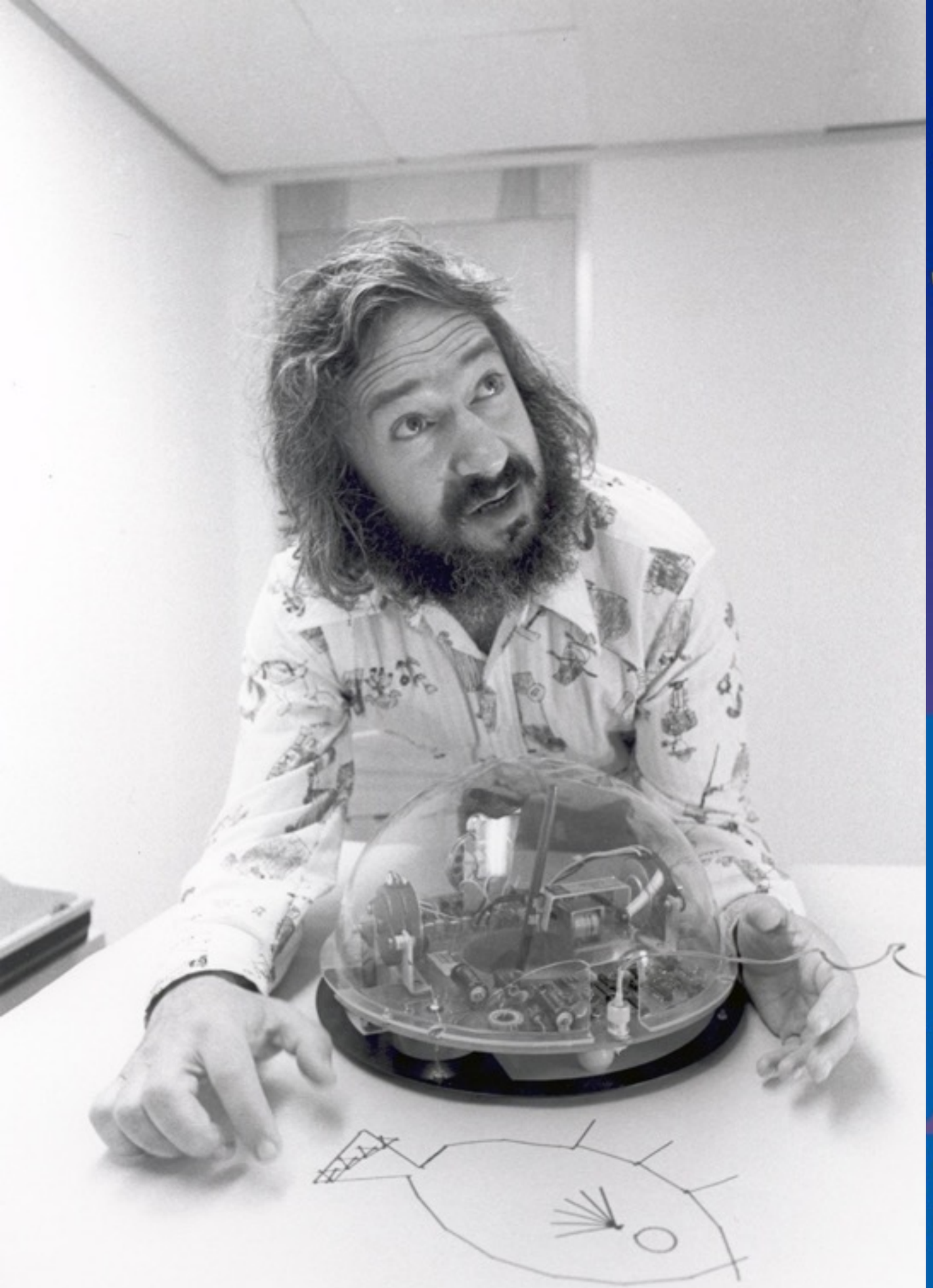
What to teach

No installer
Self-contained

Pizza

A series of microworlds





All About LOGO-
How It Was Invented and How It Works

MINDSTORMS

Children, Computers,
and Powerful Ideas

WITH AN INTRODUCTION BY JOHN SCULLEY
AND A NEW PREFACE BY THE AUTHOR

SEYMOUR PAPERT

Fixed values

```
void setup() {
    size(500, 500);
}

void draw() {
    background(255);

    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(20);
    strokeCap(SQUARE);
    arc(200, 200, 250, 250, 0, QUARTER_PI);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    ellipse(270, 230, 20, 20);
}
```

Functions: parametric values

```
void setup() {
    size(500, 500);
}

void draw() {
    background(255);
    pizzaSlice(200, 200, 250, 250, 0, QUARTER_PI);
}

void pizzaSlice(float x, float y, int w, int h, float start, float end) {
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(20);
    strokeCap(SQUARE);
    arc(x, y, w, h, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    ellipse(270, 230, 20, 20);
}
```

Oh...

```
void pizzaSlice(float x, float y, int w,
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(20);
    strokeCap(SQUARE);
    arc(x, y, w, h, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    ellipse(270, 230, 20, 20);
}
```

Relative values

```
void pizzaSlice(float x, float y, int w,
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(20);
    strokeCap(SQUARE);
    arc(x, y, w, h, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    ellipse(x + 80, y + 30, 20, 20);
}
```

Oh...

```
void setup() {
    size(500, 500);
}

void draw() {
    background(255);
    pizzaSlice(50, 250, 100, 100, 0, QUARTER_PI);
}

void pizzaSlice(float x, float y, int w, int h, flo
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(20);
    strokeCap(SQUARE);
    arc(x, y, w, h, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
```

Scaling

```
void draw() {
    background(255);
    pizzaSlice(50, 250, 100, 100, 0, QUARTER_PI);
}

void pizzaSlice(float x, float y, int w, int h, float
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(20);
    strokeCap(SQUARE);
    arc(x, y, w, h, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    ellipse(x + (w * 0.25), y + (h * 0.15), 20, 20);
}
```

Abstraction: APIs

```
void setup() {
    size(500, 500);
}

void draw() {
    background(255);
    pizzaSlice(50, 250, 150, 0, QUARTER_PI);
}

void pizzaSlice(float x, float y, float diameter, float start,
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
    arc(x, y, diameter, diameter, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    float pDiameter = diameter * 0.1;
    ellipse(x + (diameter * 0.25), y + (diameter * 0.15), pDiam
}
```

Invisible code

```
void draw() {
    background(255);

    for (int i = 0; i < 5; i++) {
        pizzaSlice(i * 100, 100, 150, 0, QUARTER_PI);
    }
}

void pizzaSlice(float x, float y, float diameter, floa
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
    arc(x, y, diameter, diameter, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    float pDiameter = diameter * 0.1;
    ellipse(x + (diameter * 0.25), y + (diameter * 0.1
```

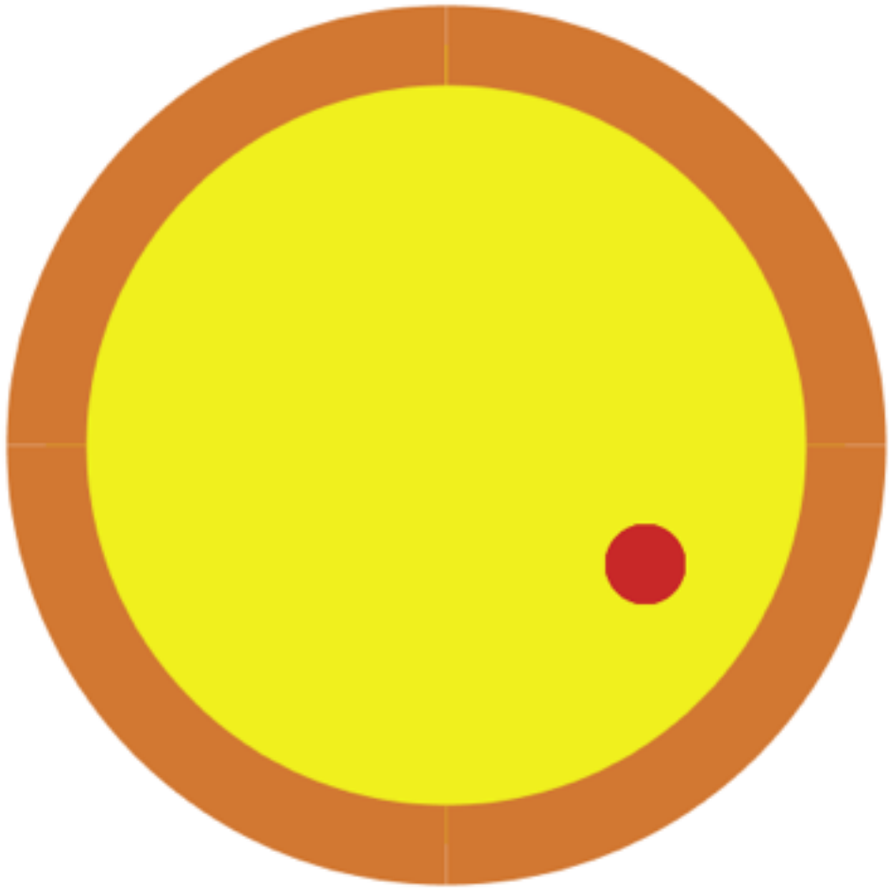
Unroll the loop

i	x
0	0
1	100
2	200
3	300
4	400

i	x
0	100
1	200
2	300
3	400
4	500

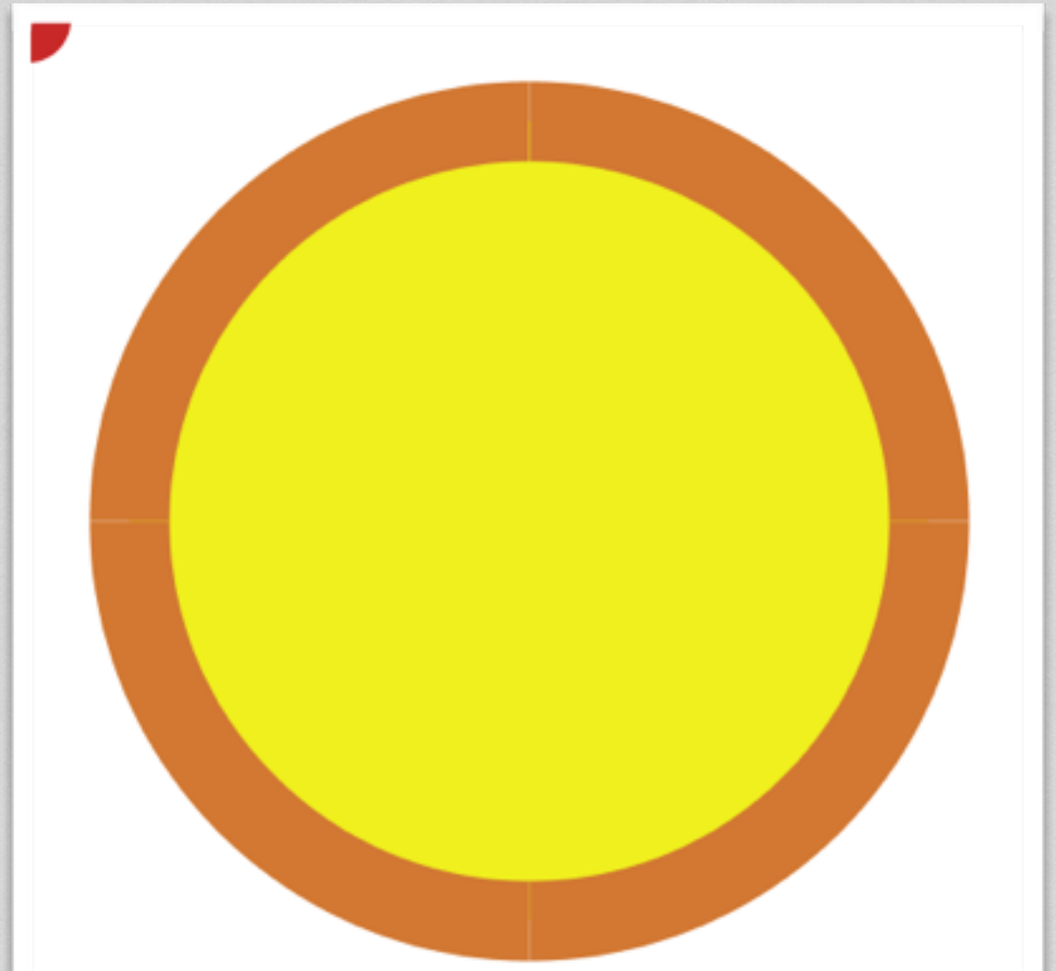
More abstraction

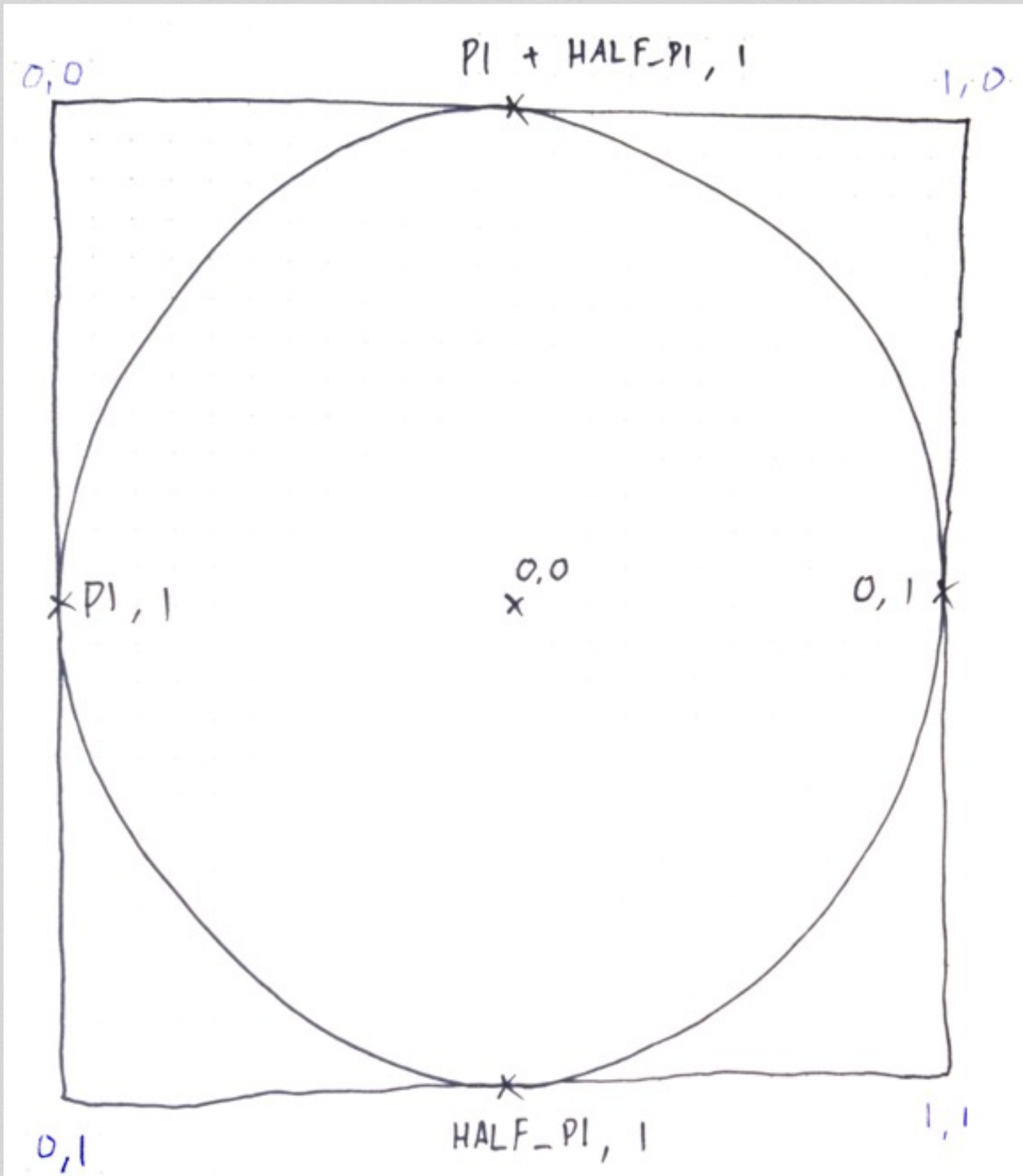
```
void setup() {
    size(500, 500);
}

void draw() {
    background(255);
    pizza(250, 250, 400, 4);
}

void pizza(float x, float y, float diameter, int numSlices) {
    float step = TWO_PI / numSlices;
    for (int i = 0; i < numSlices; i++) {
        float start = i * step;
        float end = start + step;
        pizzaSlice(x, y, diameter, start, end);
    }
}

void pizzaSlice(float x, float y, float diameter, float start,
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
```



Practical trigonometry

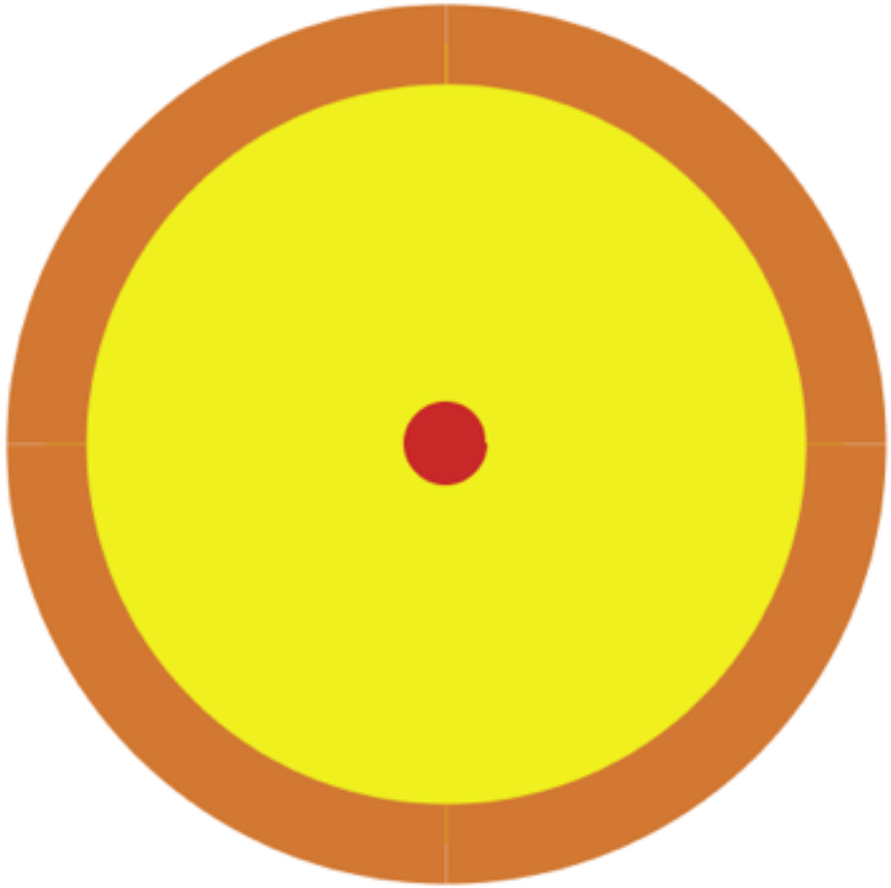
```
void pizzaSlice(float x, float y, float diameter
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
    arc(x, y, diameter, diameter, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    float px = cos(start);
    float py = sin(start);
    println(px, py);
    float pDiameter = diameter * 0.1;
    ellipse(px, py, pDiameter, pDiameter);
}
```




PI + HALF_PI, 1
0,0
1,0
PI, 1
0,0
0, 1
HALF_PI, 1
0,1
1,1

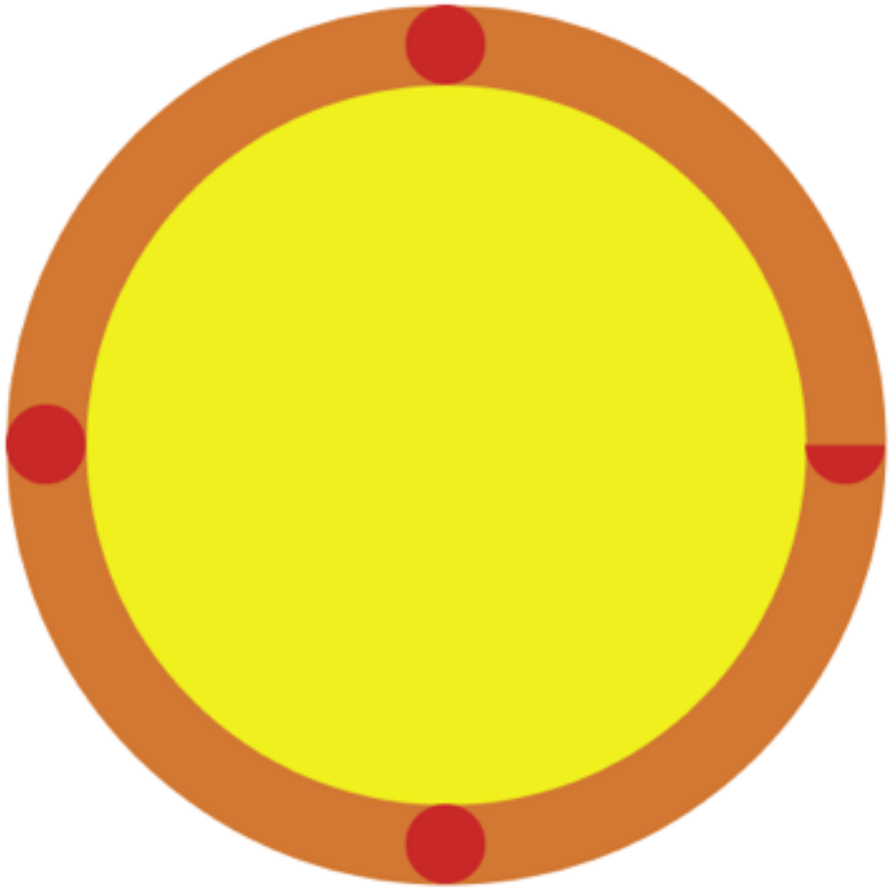
Practical trigonometry

```
void pizzaSlice(float x, float y, float diameter
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
    arc(x, y, diameter, diameter, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    float px = x + cos(start);
    float py = y + sin(start);
    println(px, py);
    float pDiameter = diameter * 0.1;
    ellipse(px, py, pDiameter, pDiameter);
}
```



Practical trigonometry

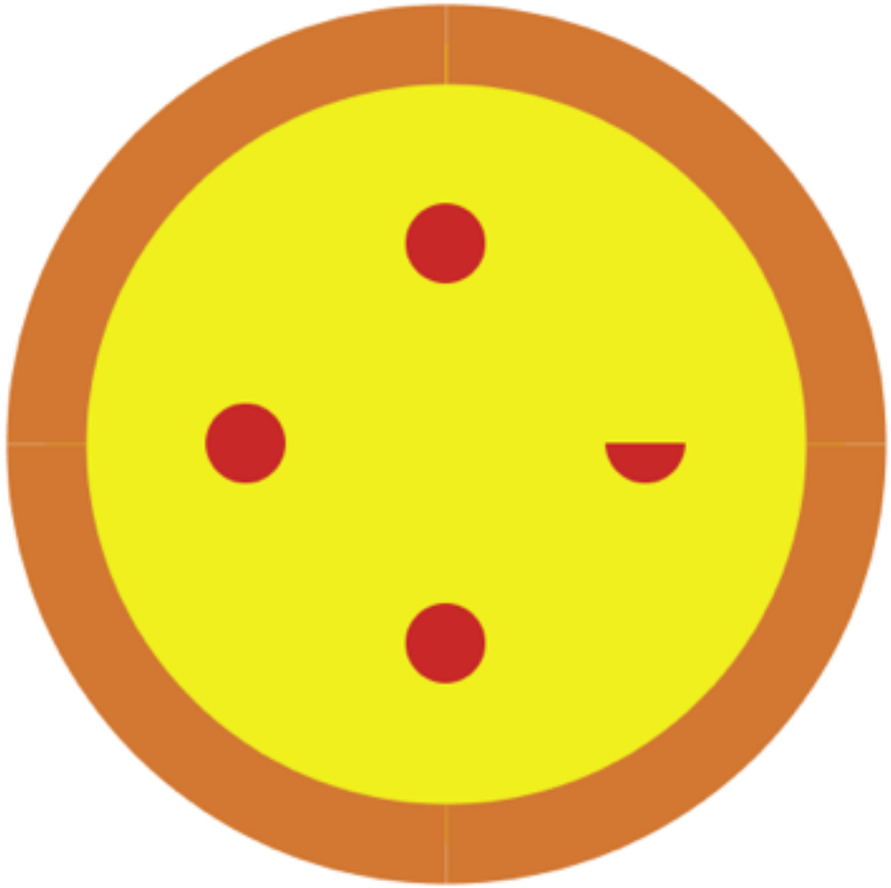
```
void pizzaSlice(float x, float y, float diameter
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
    arc(x, y, diameter, diameter, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    float radius = diameter / 2;
    float px = x + cos(start) * radius;
    float py = y + sin(start) * radius;
    println(px, py);
    float pDiameter = diameter * 0.1;
    ellipse(px, py, pDiameter, pDiameter);
}
```



Workin' on it...

Practical trigonometry

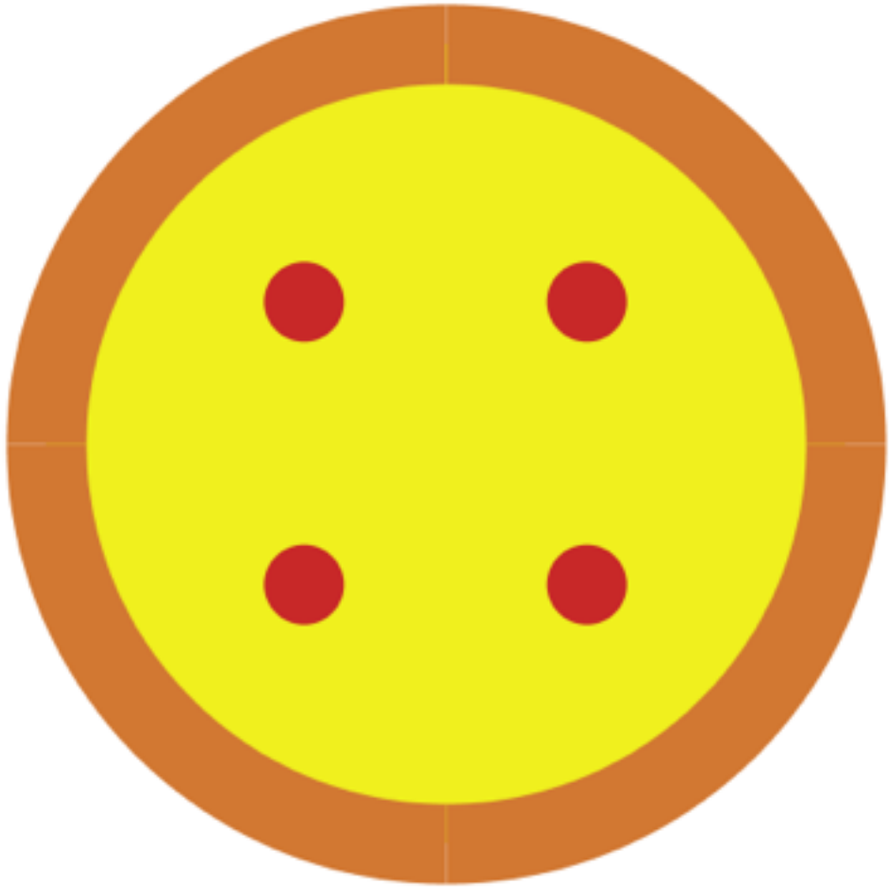
```
void pizzaSlice(float x, float y, float diameter
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
    arc(x, y, diameter, diameter, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    float radius = diameter / 2;
    float px = x + cos(start) * radius * 0.5;
    float py = y + sin(start) * radius * 0.5;
    float pDiameter = diameter * 0.1;
    ellipse(px, py, pDiameter, pDiameter);
}
```



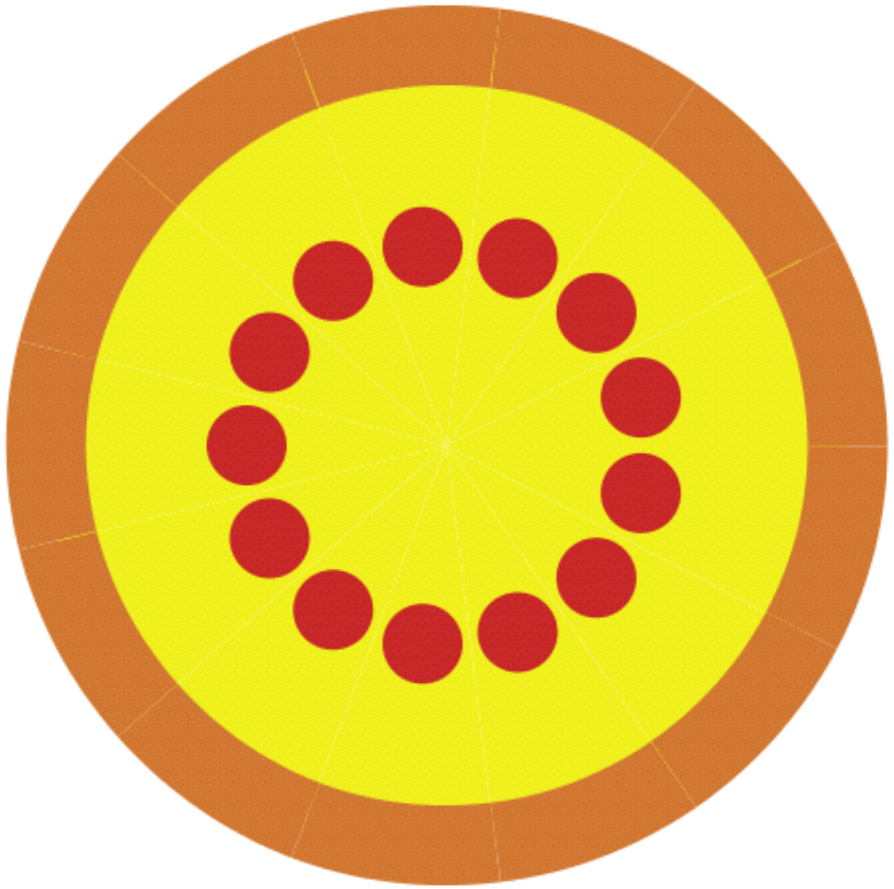
Workin' on it...

Practical trigonometry

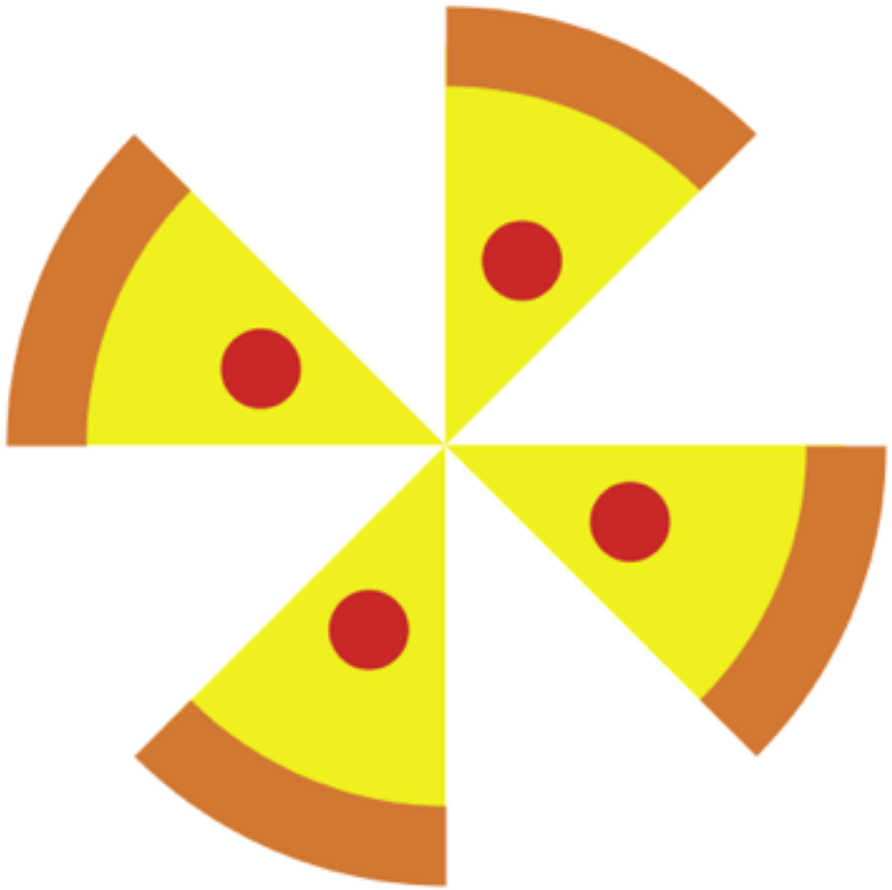
```
void pizzaSlice(float x, float y, float diameter
    // pizza
    fill(240, 240, 30);
    stroke(210, 120, 50);
    strokeWeight(diameter * 0.1);
    strokeCap(SQUARE);
    arc(x, y, diameter, diameter, start, end);

    // pepperoni
    fill(200, 40, 40);
    noStroke();
    float radius = diameter / 2;
    float pAngle = start + (end - start) * 0.5;
    float px = x + cos(pAngle) * radius * 0.5;
    float py = y + sin(pAngle) * radius * 0.5;
    float pDiameter = diameter * 0.1;
    ellipse(px, py, pDiameter, pDiameter);
}
```



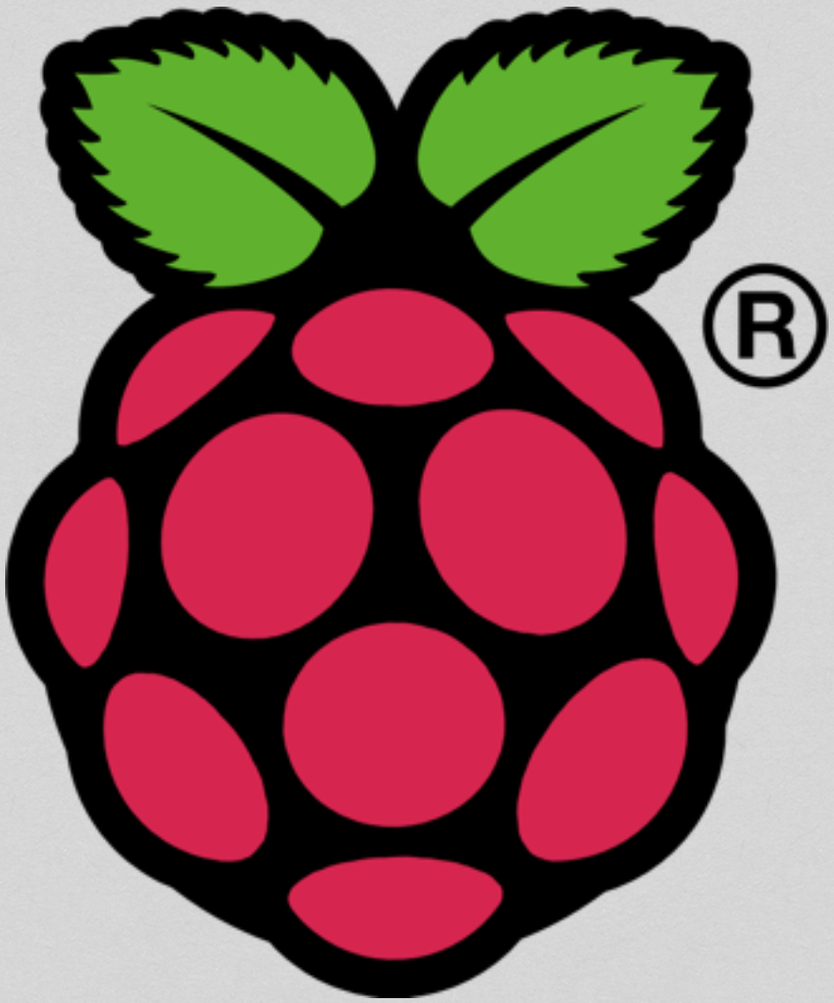
Delicious



```
pizza(250, 250, 400, 13);
```



```
if (i % 2 == 0) {
    float start = i * step;
    float end = start + step;
    pizzaSlice(x, y, diameter, start, end);
}
```



Real world

Full system

Pre-installed

Package manager


POKÉMON


```python
import random

class Pokemon():
    MAX_MOVES = 4

    def __init__(self, name, level=1):
        self.name = name
        self.level = level
        self.hp = 100
        self.poketype = None
        self.moves = []

    def attack(self, target):
        damage = random.randint(5, 40)
        target.be_attacked(damage)

    def be_attacked(self, damage):
        self.hp -= damage
        if self.hp <= 0:
            self.faint()

    def evolve(self):
        print("What's this? {} is evolving!".format(self.name))
        return self.evolves_into(self.name, self.level)

    def faint(self):
        print("{}: faint".format(self.name))

    def learn_move(self, move):
        if len(self.moves) < Pokemon.MAX_MOVES:
            self.moves.append(move)
        else:
            print("too many moves!")
```

Deep understanding

Make decisions

Add features themselves

Initially simple,
develops complexity


```python
from pokemon import Pokemon

class GrassPokemon(Pokemon):
    poketype = "grass"
    def __init__(self, name, level=1):
        super().__init__(name, level)

class Venusaur(GrassPokemon):
    evolves_into = None
    def __init__(self, name, level=1):
        super().__init__(name, level)

class Ivysaur(GrassPokemon):
    evolves_into = Venusaur
    def __init__(self, name, level=1):
        super().__init__(name, level)

class Bulbasaur(GrassPokemon):
    poketype = "grass"
    evolves_into = Ivysaur

    def __init__(self, name, level=1):
        super().__init__(name)

    def get_type(self):
        print(self.poketype)
```

Evolution & Inheritance

Is a GrassPokemon

Has a evolves_into property

CLSB Python

File Edit View Insert Format Tools Table Add-ons Help Accessibility All changes save... Comments

100% Normal text Source Co... 11 More

the move from the list of moves

06-03-15

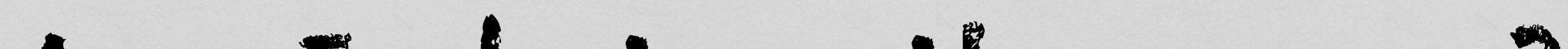
Fill out the methods in the Pokemon class:

-attack
-be_attacked

sketchpad.cc

p5js.org

codecademy.com



Am I doing it wrong?

Sonic Pi

Java

extempore


CLASS

(Obligatory academic cred slide)

Cognitive Differences Between Procedural Programming and Object Oriented Programming

Garry White, Marcos Sivitanides

Information Technology and Management
October 2005, Volume 6, Issue 4, pp 333-350

```
Buffer.read(s, "~/audio/10 hour pokemon theme.wav");
```

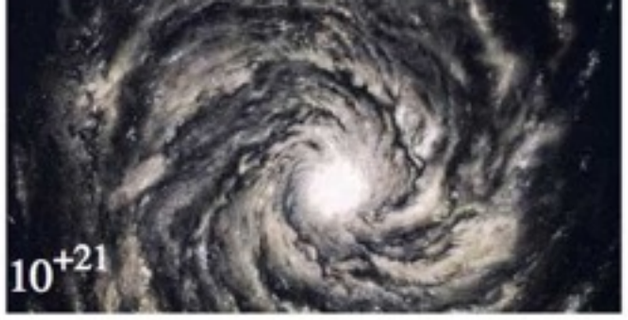
```
#include "maxiSample.h"

bool maxiSample::read(string myPath)
{
    bool result;
    fstream inFile( myPath.c_str(), ios::in | ios::binary);
    result = inFile;

    if (inFile)
    {
        //printf("Reading wav file...\n"); // for debugging only

        inFile.seekg(4, ios::beg);
        inFile.read( (char*) &myChunkSize, 4 ); // read the ChunkSize

        inFile.seekg(16, ios::beg);
        inFile.read( (char*) &mySubChunk1Size, 4 ); // read the SubChunk1Size

        //inFile.seekg(20, ios::beg);
        inFile.read( (char*) &myFormat, sizeof(short) ); // read the file format.  This should be 1 for PCM

        //inFile.seekg(22, ios::beg);
        inFile.read( (char*) &myChannels, sizeof(short) ); // read the # of channels (1 or 2)

        //inFile.seekg(24, ios::beg);
        inFile.read( (char*) &mySampleRate, sizeof(int) ); // read the samplerate

        //inFile.seekg(28, ios::beg);
        inFile.read( (char*) &myByteRate, sizeof(int) ); // read the byterate

        //inFile.seekg(32, ios::beg);
        inFile.read( (char*) &myBlockAlign, sizeof(short) ); // read the blockalign

        //inFile.seekg(34, ios::beg);
        inFile.read( (char*) &myBitsPerSample, sizeof(short) ); // read the bitspersample

        //ignore any extra chunks
        char chunkID[5]="";
        int filePos = 36;
        bool found = false;
        while(!found && !inFile.eof())
        {
            inFile.seekg(filePos, ios::beg);
            inFile.read((char*) &chunkID, sizeof(char) * 4);
            inFile.seekg(filePos + 4, ios::beg);
            inFile.read( (char*) &myDataSize, sizeof(int) ); // read the size of the data
            filePos += 8;
            chunkID[4] = '\0';
            if (strcmp(chunkID,"data") == 0)
            {
                found = true;
            }
            else
            {
                filePos += myDataSize;
            }

        }

        // read the data chunk
        printf("Data size: %d, filePos: %d\n",myDataSize, filePos);
        myData = (char*) malloc(myDataSize * sizeof(char));

        inFile.seekg(filePos, ios::beg);
        inFile.read(myData, myDataSize);
        length=myDataSize*(0.5/myChannels);
        inFile.close(); // close the input file



        floatData = (float*) malloc(length * myChannels * sizeof(float));
        vDSP_vflt16((short*)myData, 1, floatData, 1, length * myChannels);
        float divFactor=32767.0;
        vDSP_vsdiv(floatData, 1, &divFactor, floatData, 1, length * myChannels);

    }

    return result; // this should probably be something more descriptive
}
```

```
6.isEven();

6 % 2 == 0;
```

```
ofToString(6);
```

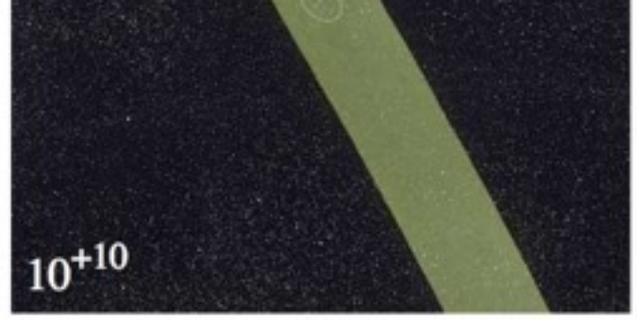
```
6.isMultipleOf(3);
```

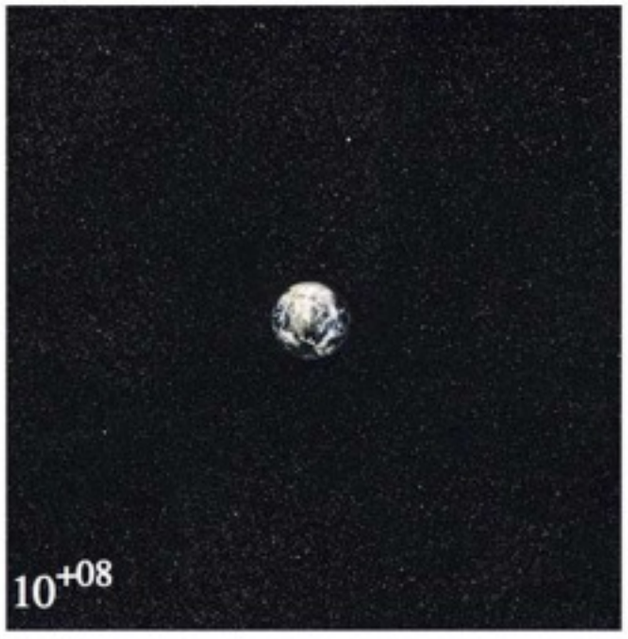
```
template <class T>
string ofToString(const T& value){
    ostringstream out;
    out << value;
    return out.str();
}
```

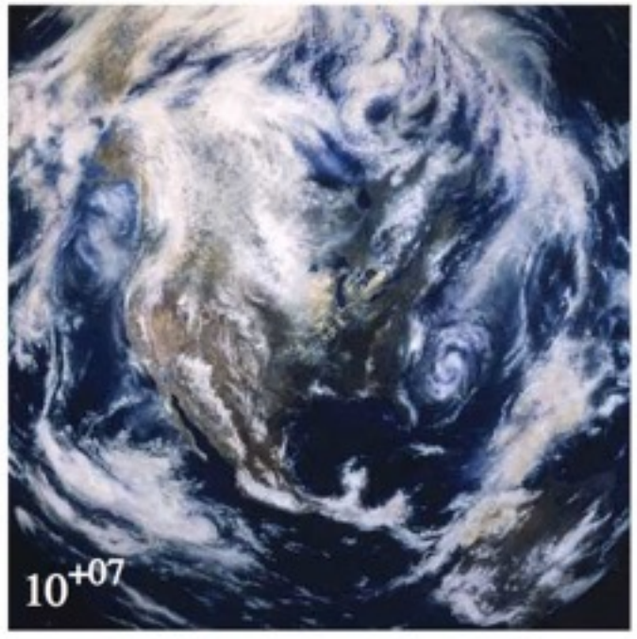

10^{+21}

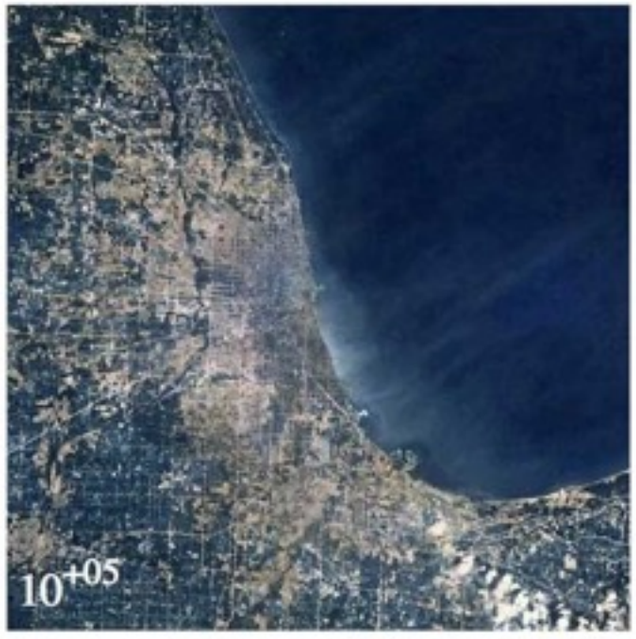

10^{+14}


10^{+13}


10^{+10}


10^{+08}


10^{+07}


10^{+05}

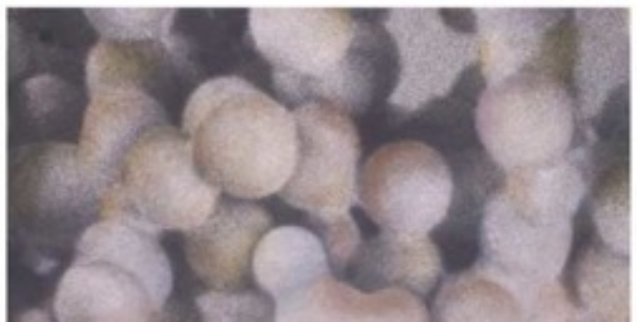

10^{+03}

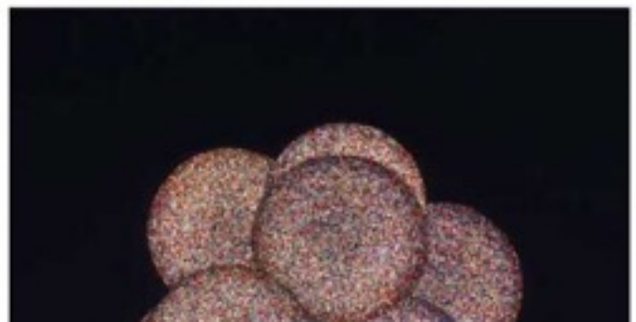

10^{+01}


10^{-01}


10^{-02}

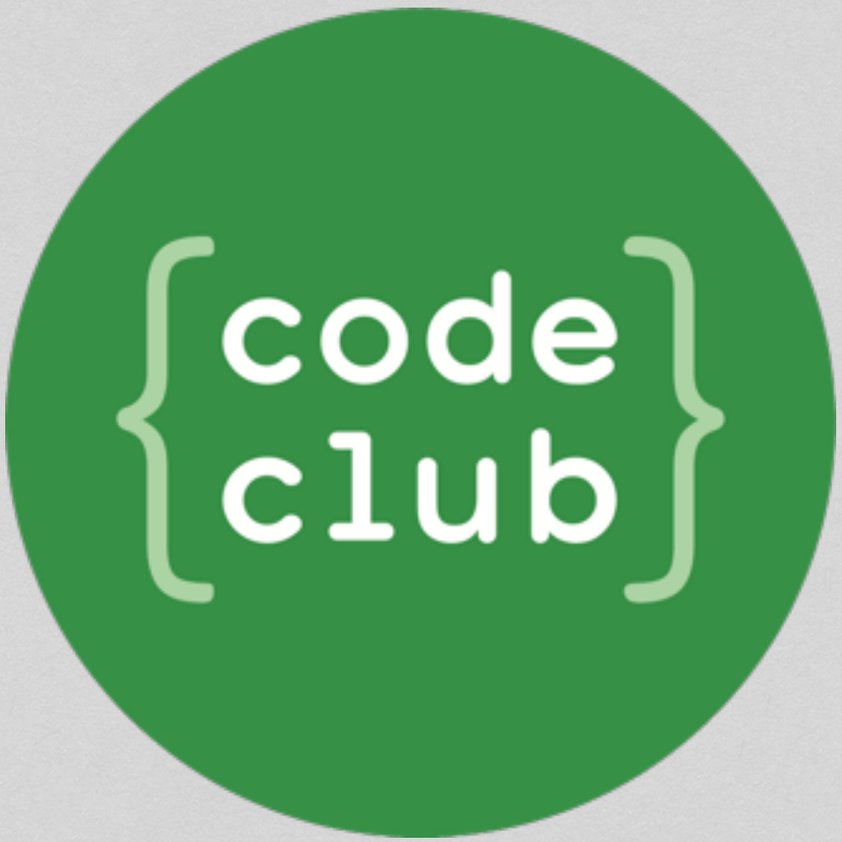

10^{-05}





Where to teach?

codeclubworld.org



Thank you!

Arthur Carabott

April 2015

@acarabott
arthurcarabott.com