Code For China
Lesson 1


1.
Intros

Icebreakers

Jessica

Computer Science Instructor

Matt

Computer Science Instructor

Cat

Design Thinking Instructor

Sophia

Design Thinking Instructor

Intros

- Name
- Favorite Hobby
- What are you excited about?
- What are you nervous about?

Name Games

Classroom Norms

Logistics

~Our Goals for the Summer~

Roadmap for today

1	2	3	4	5
Intros	CS in our daily lives	Big Ideas	Turtle	Variables, For Loops, Colors, Coordinates

2.
CS in our daily lives

Face Recognition

Virtual Reality

Self-Driving Cars

Voice Assistants

3.
Big Ideas

Big Ideas

- We tell computers to do things, so we don't have to manually do them (example: add, subtract, multiply, divide)

Big Ideas

- We tell computers to do things, so we don't have to manually do them (example: add, subtract, multiply, divide)
- Computers are good at **repeating actions multiple times**

Big Ideas

- We tell computers to do things, so we don't have to manually do them (example: add, subtract, multiply, divide)
- Computers are good at **repeating actions multiple times**

```
total = 0
for x in range (10000):
  total = total + 3
print total
```

Repeating a task 10,000 times

From Code → CPU

- We talk to computers using **code**
  - Code can be in any language: Python, Java, C++

From Code → CPU

- We talk to computers using **code**
  - Code can be in any language: Python, Java, C++
- Computers can’t understand code we write directly

From Code → CPU

- We talk to computers using **code**
  - Code can be in any language: Python, Java, C++
- Computers can't understand code we write directly
- So it must be translated into machine instructions for the **CPU - Central Processing Unit**

Step 1: Code

```
total = 0
for x in range (10000):
  total = total + 3
print total
```

Step 2: Machine Instructions for CPU

```
1  main:
2    push rbp
3    mov rbp, rsp
4    mov DWORD PTR [rbp-4], 0
5    mov DWORD PTR [rbp-8], 0
6  .L3:
7    cmp DWORD PTR [rbp-8], 9999
8    jg .L2
9    add DWORD PTR [rbp-4], 3
10   add DWORD PTR [rbp-8], 1
11   jmp .L3
12 .L2:
13   mov eax, 0
14   pop rbp
15   ret
```

From Code → CPU

- We talk to computers using **code**
  - Code can be in any language: Python, Java, C++
- Computers can't understand code we write directly
- So it must be translated into machine instructions for the **CPU - Central Processing Unit**

Step 1: Code

```
total = 0
for x in range (10000):
  total = total + 3
print total
```

Step 2: Machine Instructions for CPU

```
1  main:
2    push rbp
3    mov rbp, rsp
4    mov DWORD PTR [rbp-4], 0
5    mov DWORD PTR [rbp-8], 0
6  .L3:
7    cmp DWORD PTR [rbp-8], 9999
8    jg .L2
9    add DWORD PTR [rbp-4], 3
10   add DWORD PTR [rbp-8], 1
11   jmp .L3
12 .L2:
13   mov eax, 0
14   pop rbp
15   ret
```

4.
Turtle

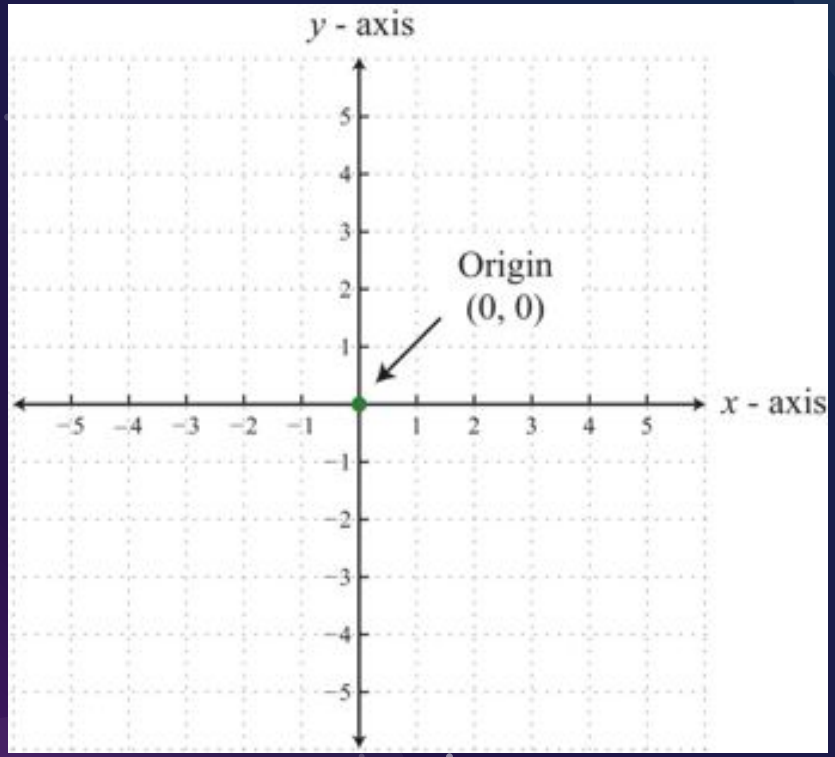
What is Turtle?

- Turtle lives in a 2D world
- Turtle knows Python
- We will use Turtle for the first few days to introduce key CS concepts

Turtle Facts

- Turtle is your robot
- Turtle can draw
  - You tell Turtle what to draw

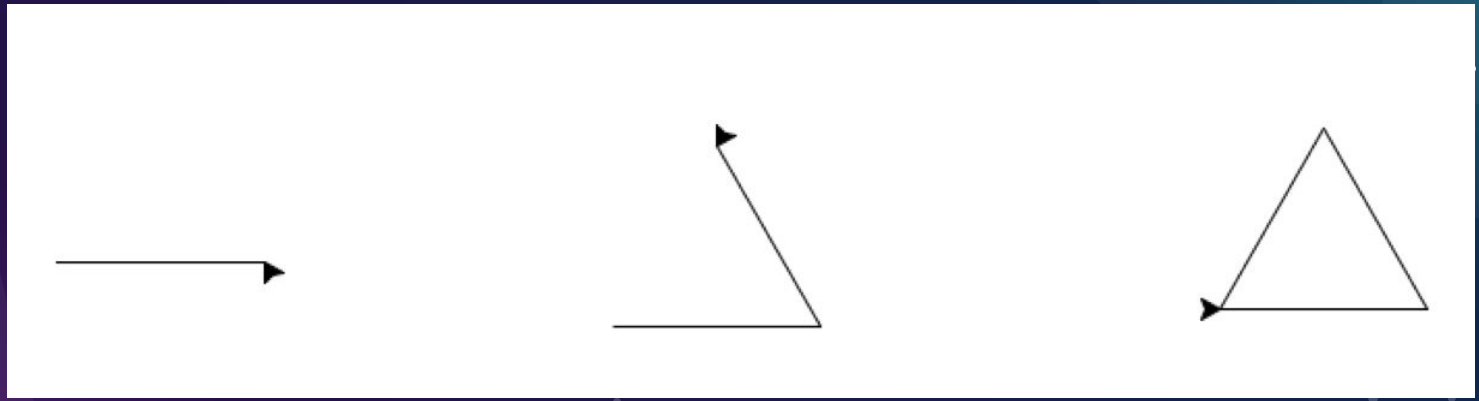
Turtle Start



Basic Turtle Program Outline

```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########


########################################
```

Example: Draw a Triangle



Example: Draw a Triangle

https://repl.it/@jessicae5/turtle-triangle-1

```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########


########################################
```

Your Turn

- Find your computer (has your name on it)
- Go to: **github.com/jessicae5/CodeForChina/blob/master/blank.py**
- Copy the code
- Open Sublime
- Paste
- Save as turtle.py on your Desktop

Your Turn: Draw a Square

```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########



########################################
```

Hint: Use these 2 commands

jess.forward()
jess.left()

CNTRL + SHIFT + B + ENTER to run

Variables

- Every variable has a **name** & **value**
- Can only have 1 value at a time
- Variables can be anything!
  - Example: 3, “Jessica”, “Matt”, 13.2456

Your Turn: How many variables?

```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########
jess.forward(100)
jess.left(90)
jess.forward(100)
jess.left(90)
jess.forward(100)
jess.left(90)
jess.forward(100)
########################################
```

Your Turn: Add Variables

- Rewrite your Square Program with Variables
- 1 variable for **distance**
- 1 variable for **angle**

For Loops

```
....
######### YOUR CODE GOES HERE #########
distance = 100
angle = 90
jess.forward(distance)
jess.left(angle)
jess.forward(distance)
jess.left(angle)
jess.forward(distance)
jess.left(angle)
jess.forward(distance)
jess.left(angle)
########################################
....
```

For Loops

```
for i in range(4):
    … code …
```

Your Turn: Add For Loop

- Rewrite your Square Program with a **For Loop**
- Remember to keep your **variables** too!

For Loops

```
....
######### YOUR CODE GOES HERE #########
distance = 100
angle = 90
for i in range(4):
    jess.forward(distance)
    jess.left(angle)
########################################
….
```

Random Numbers

- Can ask the computer for a random number
- Assign to a variable

Ages in our room from 11 to 23

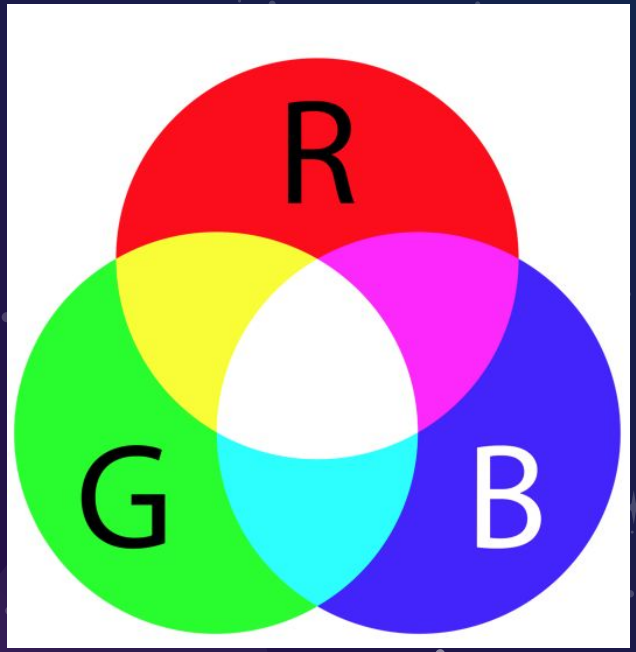
```
age = random.randint(11, 23)
```

Your Turn: Add Random Numbers

Rewrite your Square Program using **Random Numbers**

```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########
distance = 100
angle = 90
for i in range(4):
    jess.forward(distance)
    jess.left(angle)
########################################
```

RGB (Red, Green, Blue)



RGB in Turtle

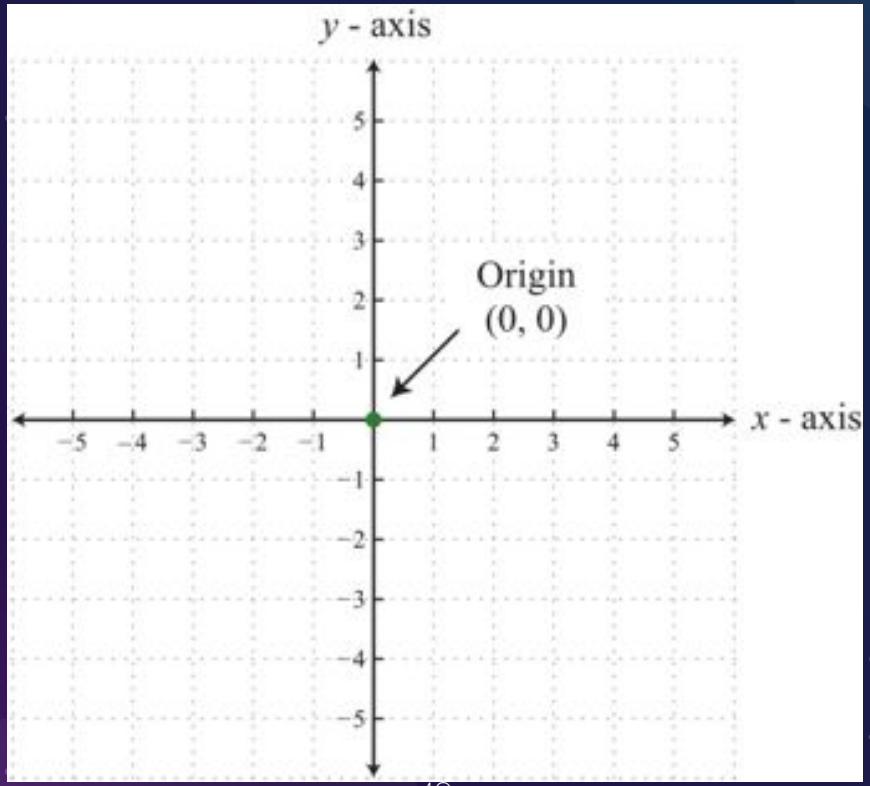
```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########

jess.color(r, g, b) <-- r, g, b are numbers between 0 and 255
```

Your Turn: Add Colors

- Rewrite your Square Program using **Colors** (use variables!)

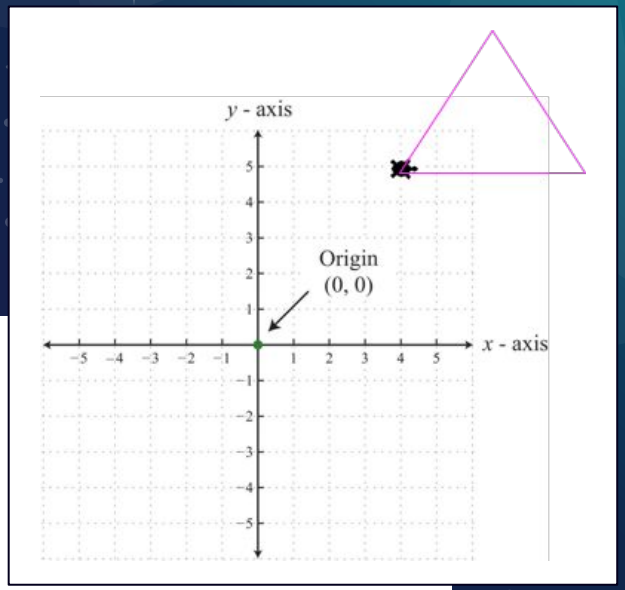
```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########
jess.color(r, g, b) <-- r, g, b are numbers between 0 and 255
distance = random.randint(10, 100)
angle=90
for i in range(4):
    jess.forward(distance)
    jess.left(angle)
```

Coordinates in Turtle



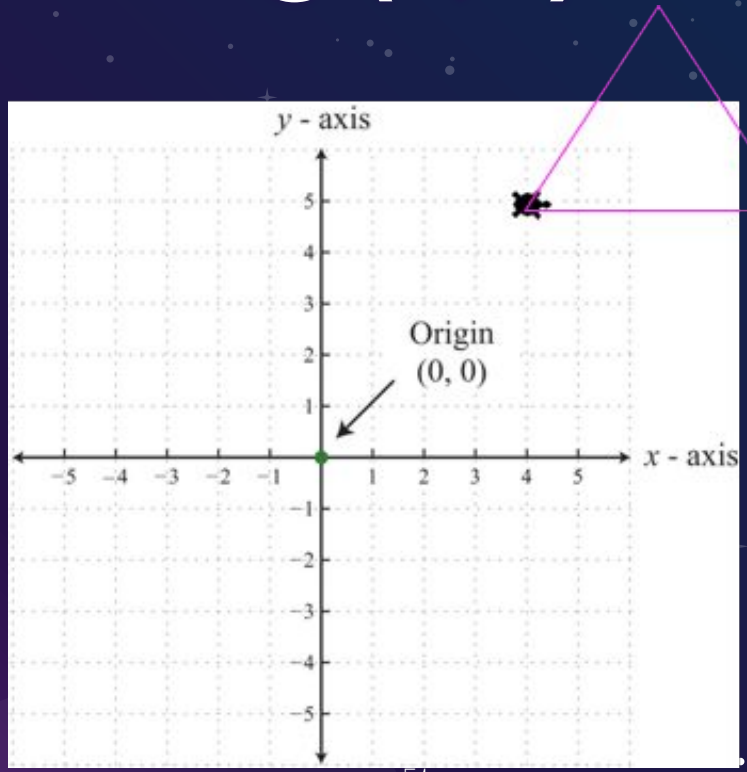
Coordinates in Turtle

- the **goto** command
- draws unless **jess.penup()**
- to draw, use **jess.pendown()**

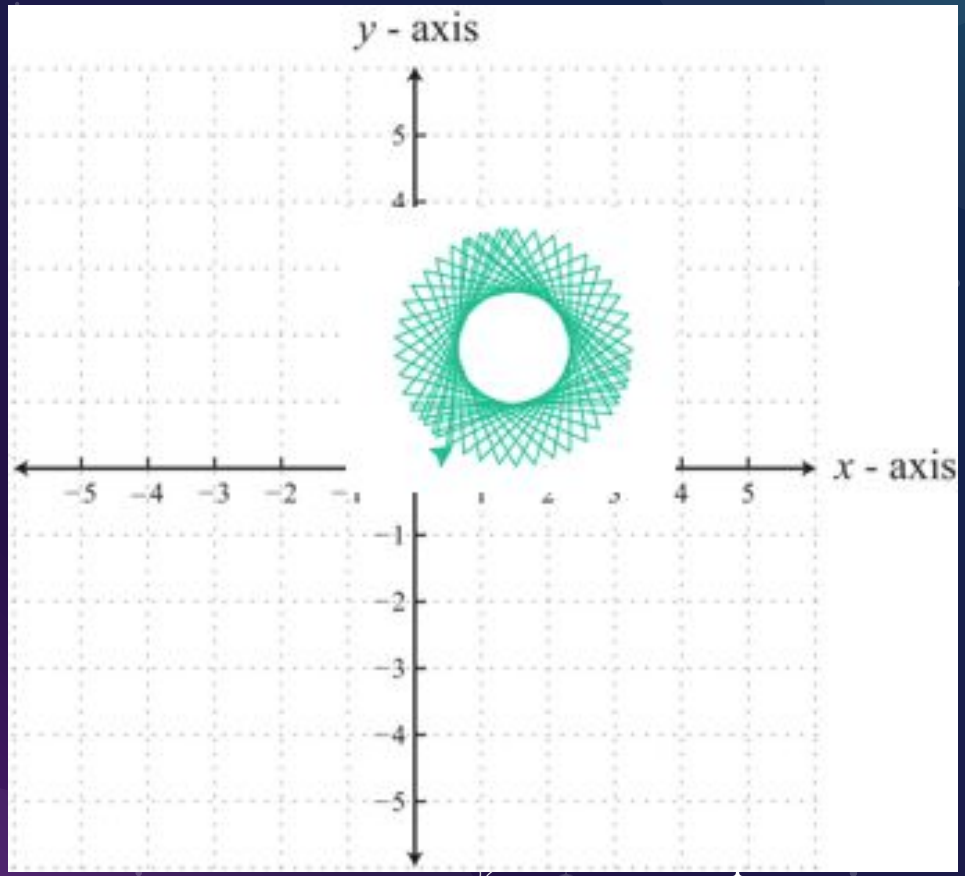



```
import turtle
window = turtle.Screen()
window.bgcolor("white")
jess = turtle.Turtle()
jess.speed(1)
######### YOUR CODE GOES HERE #########

jess.goto(x, y) <-- x, y are variables that are numbers
```

Example: Random Color/Size Triangle @ (4, 5)



Your Turn: Create This



Your Turn: Random Spirals

- Use what you learned to make Turtle **draw spirals** with
  - A random size
  - A random color
  - A random location