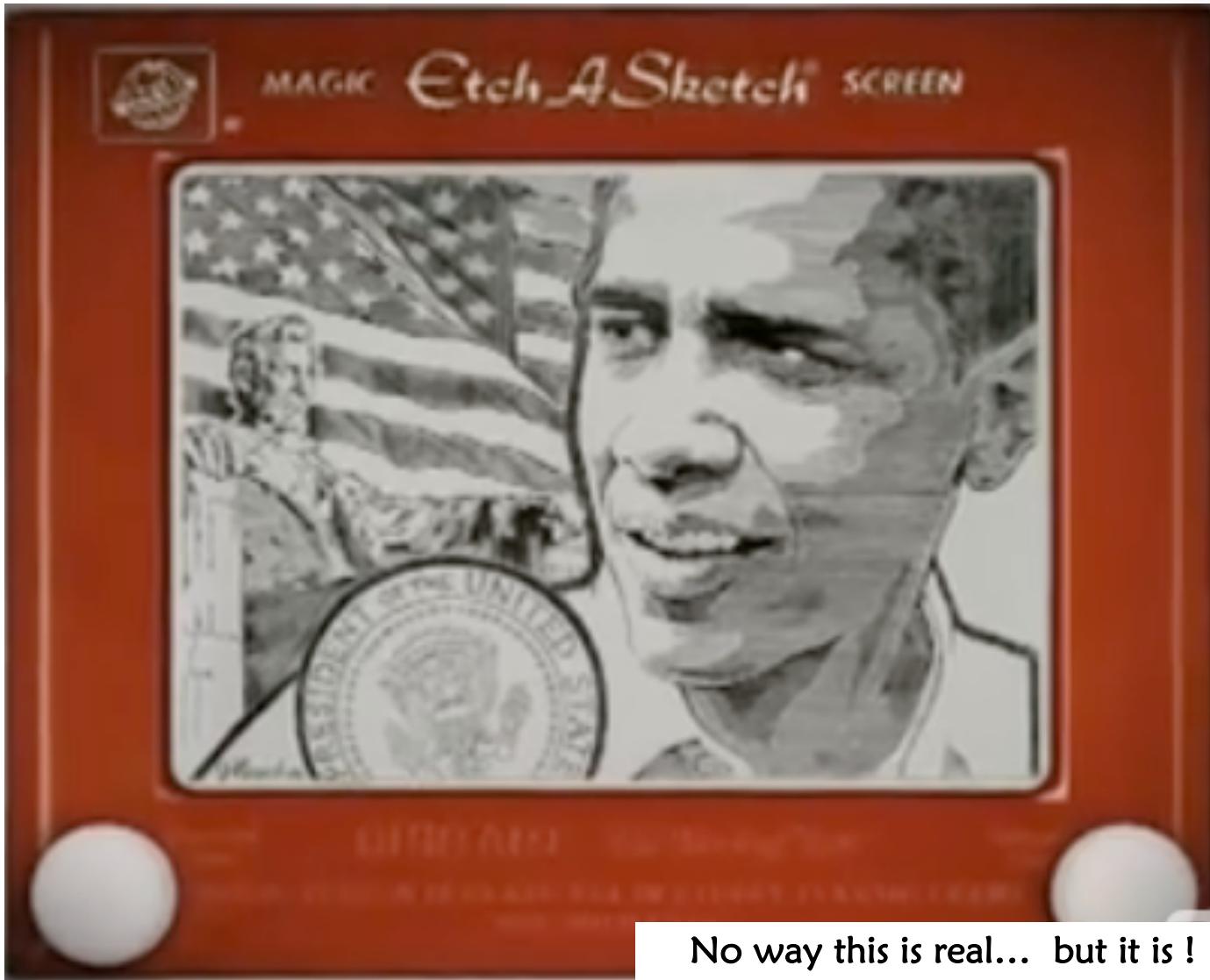


# Turtle Graphics Modules

Introduction to Computer Science!

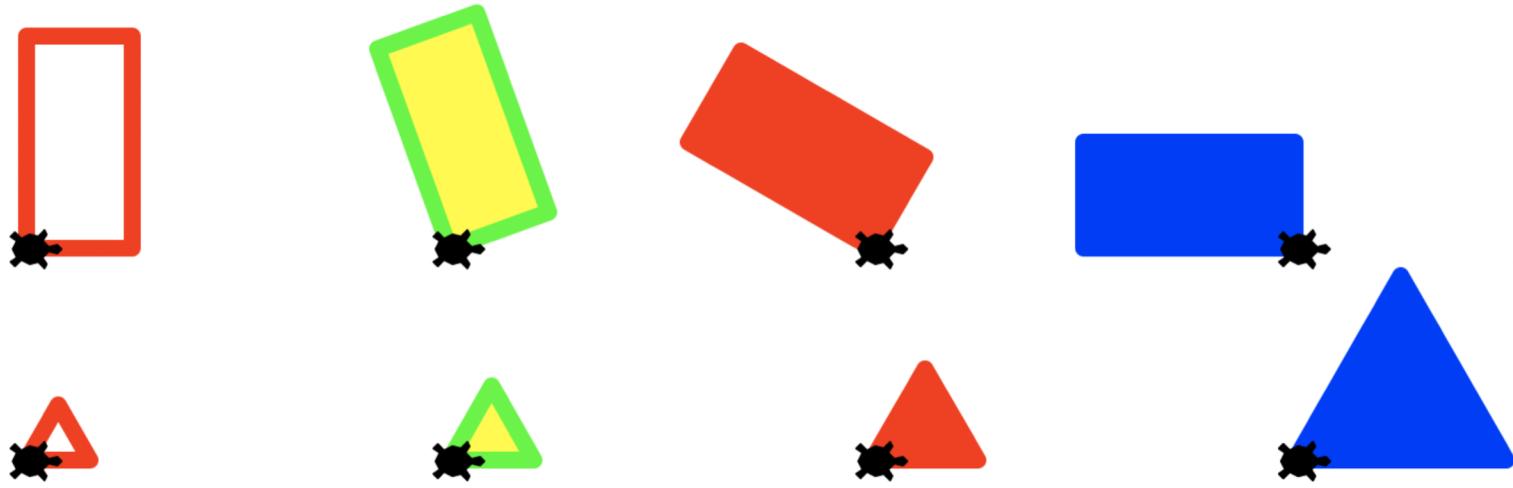


# Etch-a-Sketch ?



No way this is real... but it is !

# Lab03: Turtle Graphics



```
drawRectangle(t, width, height, tilt, penColor, fillColor)
```

```
drawTriangle(t, side, penColor, fillColor)
```

```
drawTwoRectangles(t)
```

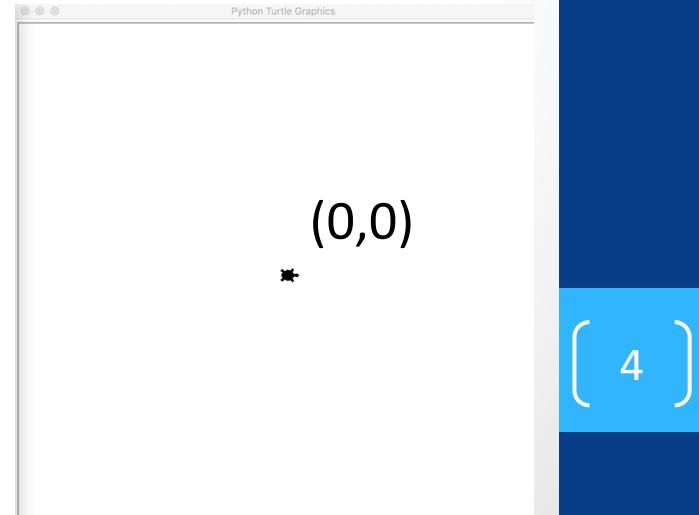
```
drawTwoTriangles(t)
```

# Turtle- getting started

```
import turtle  
# This statement allows you  
to use all the functions in  
the turtle package
```

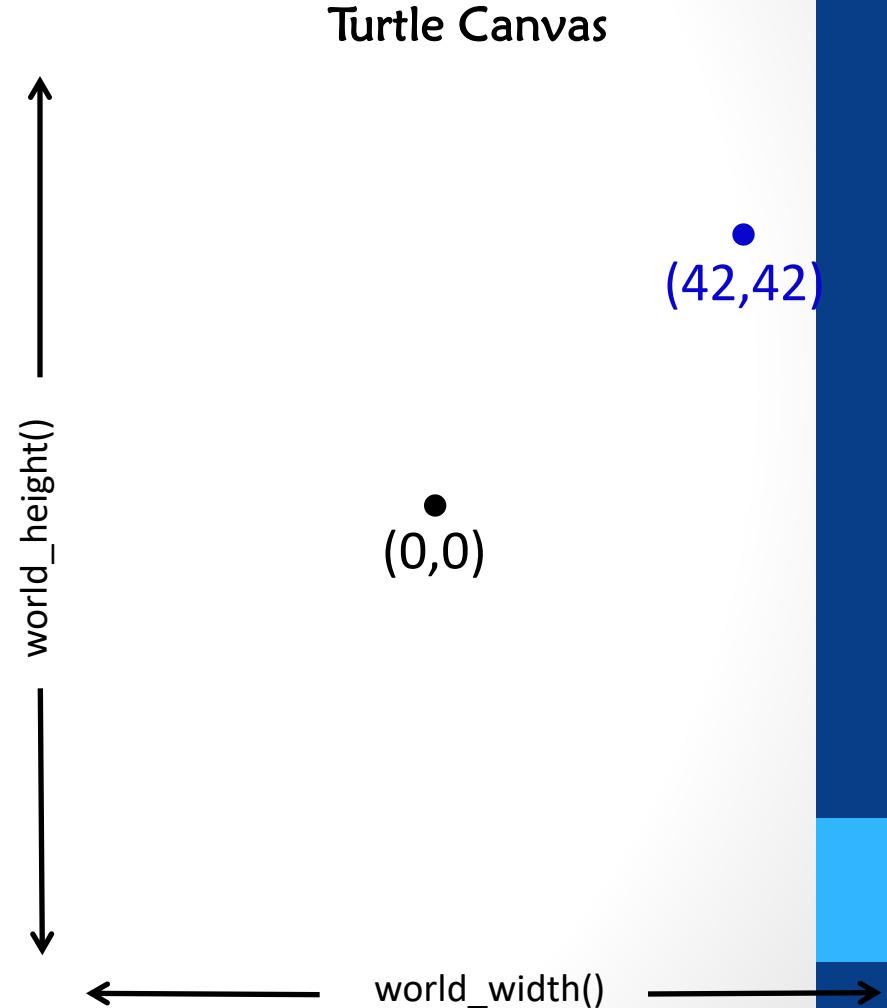
```
jane = turtle.Turtle()  
# create a new "turtle"  
object called jane
```

```
jane.shape("turtle")  
# change the shape of the  
turtle
```



# Python's Etch-a-Sketch

```
import turtle  
  
jane = turtle.Turtle()  
  
jane.forward( 100 ) 100 pixels!  
jane.left(60 ) degrees!  
jane.right(90) 90 degrees!  
jane.width(8)  
  
jane.color("green", "yellow")  
  
jane.up()  
  
jane.forward(50)  
  
jane.down()
```



# ConcepTest

Which order of instructions produces the following output:



Red: Initial position and orientation

Black: Final position and orientation

```
jane.forward(100) #(1)  
jane.left(90) #(2)  
jane.forward(50) #(3)  
jane.right(90) #(4)
```

- ▶ A. (1), (2), (3), (4)
- ▶ B. (4), (3), (2), (1)
- ▶ C. (1), (4), (2), (3)
- ▶ D. (1), (4), (3), (2)

# Writing your own module

```
# mycircle.py

# Program that draws multiple circles

import turtle

def fancyCircle(t, radius, penColor, fillColor):

    # Write a function that uses the turtle t

    # to draw a circle with a given radius

    # and color

    return

# Call the function to draw three circles

jane = turtle.Turtle()

fancyCircle(jane, 100, "red", "yellow")

fancyCircle(jane, 50, "red", "red")
```

# Writing your own module

```
# happyFace.py

# Uses the mycircle.py as a module

import turtle

import mycircle

def happyFace(t) :

    # Use the fancyCircle from mycircle

    # to draw a face, eyes, so on

    # Draw the face

    mycircle.fancyCircle(t, 100, "yellow", "yellow")

    # More code here

    return

# Call the function to draw three circles

jane = turtle.Turtle()

happyFace(jane)
```

# Writing your own module

```
# mycircle.py

# Program that draws multiple circles

import turtle

def fancyCircle(t, radius, penColor, fillColor):

    # Write a function that uses the turtle t

    # to draw a circle with a given radius

    # and color

    return

# To run the following lines conditionally, use the

# if __name__=="__main__": clause

if __name__=="__main__":

    jane = turtle.Turtle()

    fancyCircle(jane, 100, "red", "yellow")

    fancyCircle(jane, 50, "red", "red")
```

# Loops:

# Repetition without being repetitive

## Syntax

```
for <item> in <collection>:  
    # Code to loop over  
    print("Repeat this")
```

## Example: Iterating through collections

- Print each character of a string
- Print each element of a list
- Print each element of a tuple

# Concept Test

- What is the output of this code?

```
for x in [1, 2, 3]:  
    print('Hello'*x) # using x inside the loop
```

- A. 1 2 3
- B. 'Hello' is printed 3 times
- C. Hello  
HelloHello  
HelloHelloHello
- D. None of the above

# Range() function

- Used in a loop when we know the number of times we want to repeat executing some code

```
range(5)      # think of it as producing a list [0, 1, 2, 3, 4]
range(1, 5) # The first parameter is a starting value
            # The second parameter is the stopping value
            # Output [1, 2, 3, 4]
range(0, 10, 2) # The third parameter is the step count
                 #[0, 2, 4, 6, 8]

for x in range(5):
    print('Hello')
```

# Concept Test

What is the output of this code?

```
for x in range(1,4,2):  
    print(2**x, end = " ")
```

- A. 1 4 2
- B. 2 16 4
- C. 2 8 16
- D. 2 8
- E. None of the above