Code in Place 2023

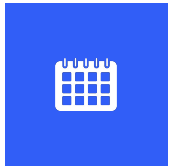
Stanford CS106A

Section - Week 4

Functions & Graphics

David Tsai, Code in Place

Today's Agenda



1. Check-In

How are we all doing?

2. Concepts Review

Functions, Graphics

3. Practice Problem

"Draw Random Circles"

Before We Start

How are you all doing?
Hopefully your third week of CIP went well!

- Is there anything blocking you from completing Week 1-3 assignments?
- Any major questions?

Concepts Review

Functions: Variable Scope

Variable scope

- When you define a variable within a function, it will have **local scope** (its scope lies ONLY within the function).
  - These types of variables are called **local variables**.
- Local variables exist only for as long as the function is running.
- Local variables cannot be changed or accessed from outside the function.

```
def greet():
      message = “Hello”       # Local variable within the greet() function.
      print(message)          # Print the local variable.

def main():
      print(message)          # Error!  The variable “message” does not exist!
```

Functions: Passing Data

Arguments & Parameters

- Pass data from a calling function to a helper function.

`return` statement

- Return data from helper function to calling function.

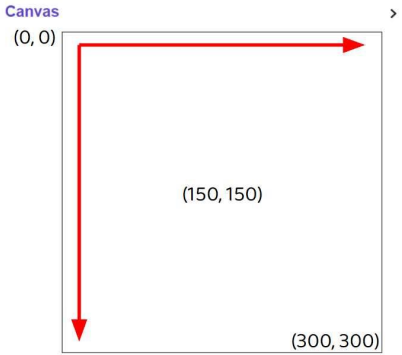
```
def main():
      mid = average(7.2, 31.5)      # Function call.  Parameters: 7.2, 31.5
      print(mid)

def average(a, b):                  # Function definition.  Arguments: a, b
      sum = a + b
      return sum / 2                # Ends the function and gives back a value.
```

Graphics: Canvas

- We'll use Python's "Tkinter" package to draw on a **Canvas**.
  - Pixels start at position (0, 0) on the top-left corner.
  - x increases going right, y increases going down.

```
CANVAS_WIDTH = 300
CANVAS_HEIGHT = 300
def main():
      canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
```

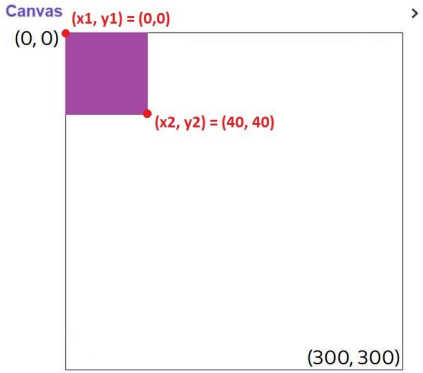


(Image source: Cameron Mohne)

Graphics: Draw Rectangle

- `canvas.create_rectangle(x1, y1, x2, y2, color)`
  - `(x1, y1)` = top-left corner
  - `(x2, y2)` = bottom-right corner

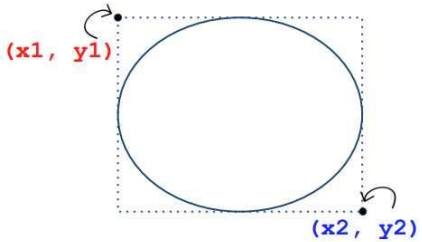
```
def main():
      canvas = Canvas(300, 300)
      canvas.create_rectangle(0, 0, 40, 40, “purple”)
```



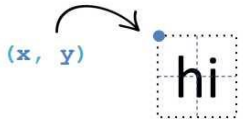
Graphics: More Shapes

- `canvas.create_line(x1, y1, x2, y2, color)`

- `canvas.create_oval(x1, y1, x2, y2, color)`



- `canvas.create_text(x, y, text = “hi”)`

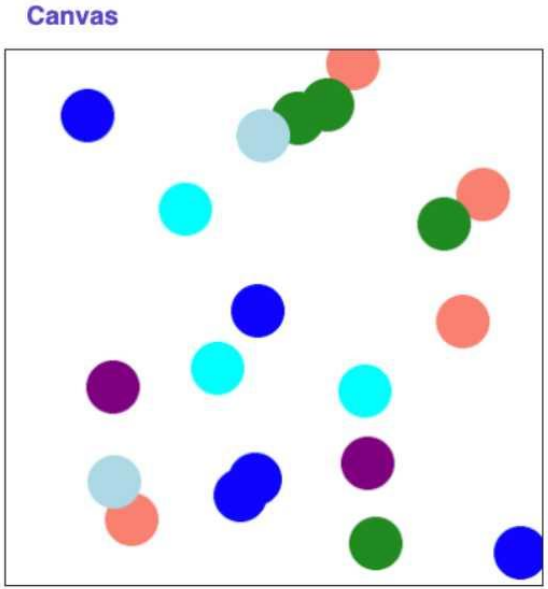


`random` Library

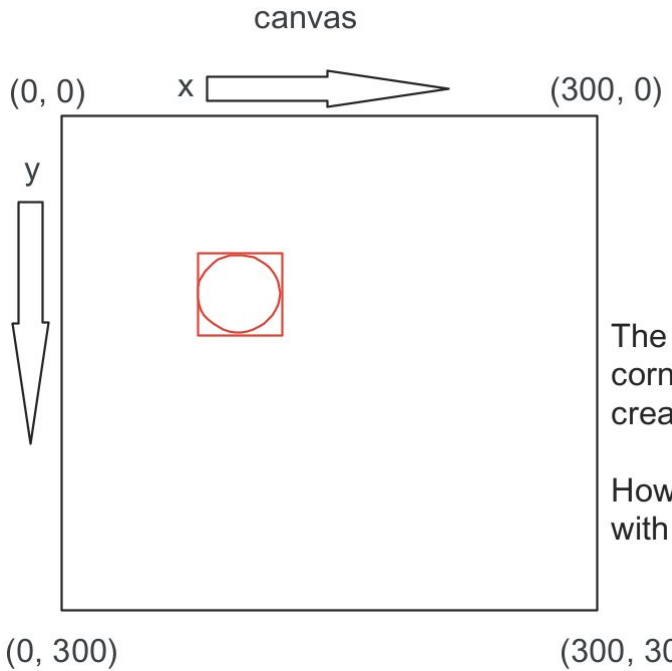
- `random.randint(min, max)`
  - Returns a random int between min and max, inclusive.
  - `dice_roll = random.randint(1, 6)  # number from 1 to 6`
- `random.uniform(min, max)`
  - Returns a random float between min and max, inclusive.
- `random.random()`
  - Returns a random float between 0.0 and 1.0, inclusive.
- `random.seed(1)`
  - Fixes the random generator so that it gives the same sequence of numbers each time.
  - Can change 1 to another number for a different fixed sequence.
  - Helpful for debugging!

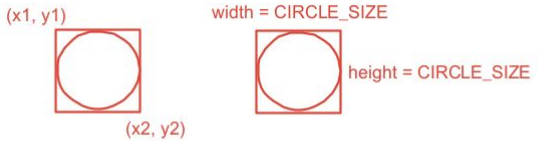
Section Exercise: “Draw Random Circles”

Draw 20 circles at random positions with random colors.



Section Exercise: “Draw Random Circles”






```
x1 = random.randint(0, CANVAS_WIDTH)
y1 = random.randint(0, CANVAS_HEIGHT)

canvas.create_oval(x1, y1, x2, y2, "red")
canvas.create_oval(x1, y1, x1 + CIRCLE_SIZE, y1 + CIRCLE_SIZE, "red")
```

The code above will randomly generate the top-left corner coordinates of a circle (x1, y1) and use them to create a circle on the canvas.

How do we ensure that the circle is always created with its entire body inside the canvas borders?

```
x1 = random.randint(0, CANVAS_WIDTH - CIRCLE_SIZE)
y1 = random.randint(0, CANVAS_HEIGHT - CIRCLE_SIZE)
```