WELCOME TO

TECHNOVATION

Week 2 - October 1

MICHIGAN STATE UNIVERSITY

Attendance Form

https://forms.gle/3BDPR9RHjyDE1MJj7

Agenda

- Club contract & Icebreaker
- Lesson 2: Colors and Variables
  - Colors
  - Variables
  - User Inputs
- Coding challenges
- Standup

Club Contract

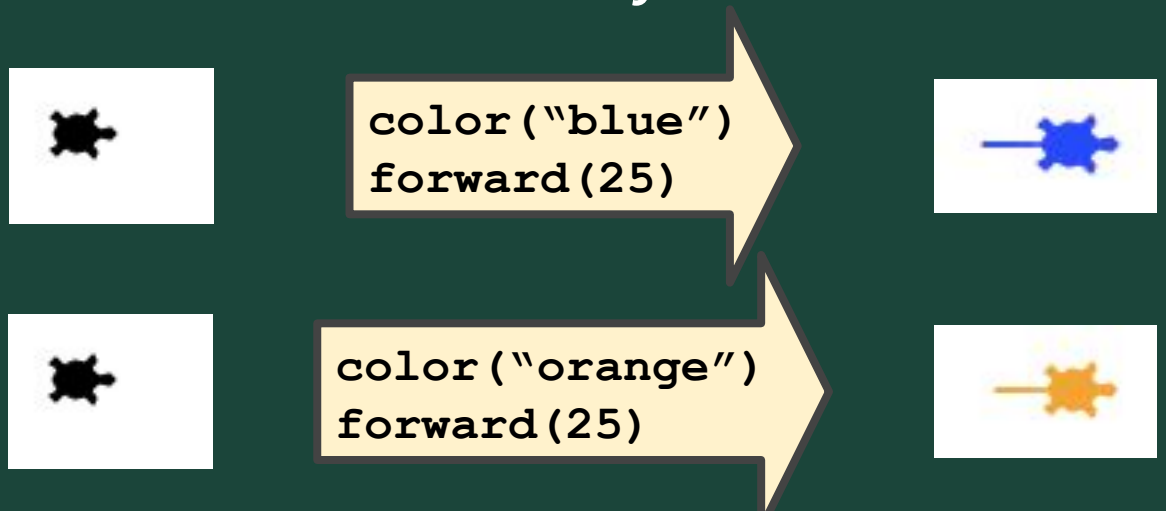
We’ve written down all of your responses from the club contract last week and now it’s time to sign it!

Ice Breaker! - Name Train

Tracy Command: color

`color("color_name")`

Changes the color of the trail Tracy leaves



Some Color Names You Can Use

"black"	"orange"	"gold"	"white"
"blue"	"pink"	"gray"	"violet"
"brown"	"purple"	"green"	"indigo"
"cyan"	"red"	"tangerine"	"mauve"

Default color: "black"

More color names at:
https://www.webucator.com/article/python-color-constants-module/

Tracy knows a lot of ‘Color Names’

For example, some shades of blue:

`"blue"`	`"midnight blue"`	`"turquoise"`
`"light blue"`	`"sky blue"`	`"cyan"`
`"medium blue"`	`"royal blue"`	`"teal"`
`"dark blue"`	`"cornflower blue"`	`"aqua"`

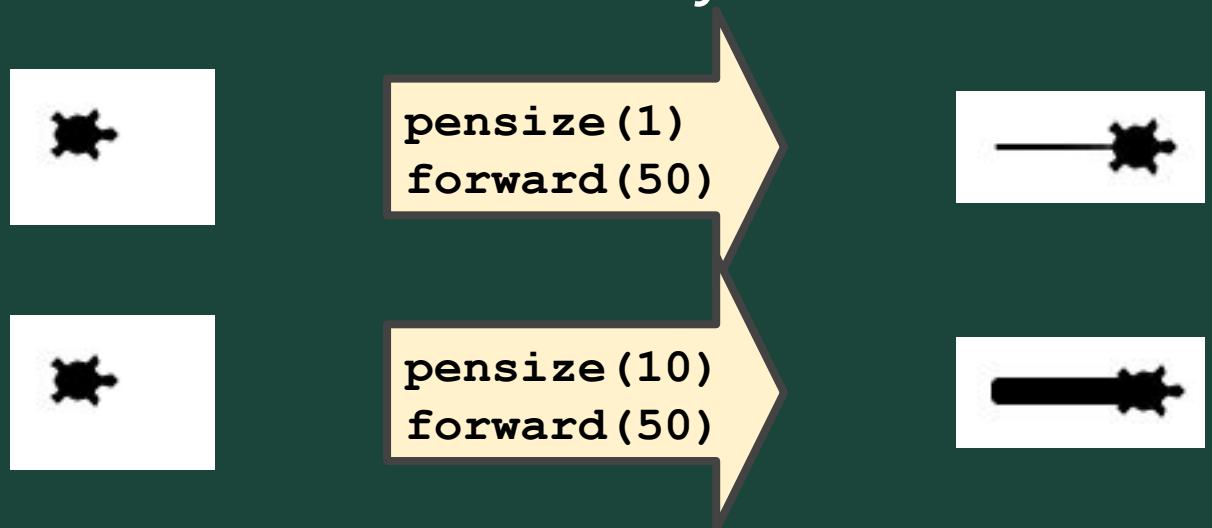
Default color: **`"black"`**

For others, see:
https://www.webucator.com/article/python-color-constants-module/

Tracy Command: pensize

`pensize(`*`number`*`)`

Changes the thickness
of the trail Tracy leaves



Tracy Command: begin_fill(), end_fill()

`begin_fill()`

Start keeping track of closed shapes that are drawn

`end_fill()`

Stop keeping track of closed shapes and fill them

```
begin_fill()
circle(50)
end_fill()
```

Tracy command: setposition

`setposition(x, y)`

Moves Tracy to the given coordinate

Tracy command: speed

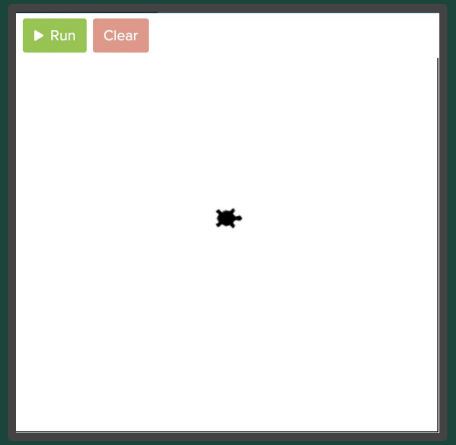
`speed(number)`

Sets how quickly Tracy will move through commands

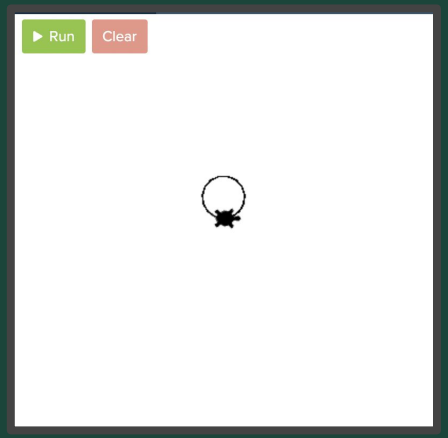
Slowest *Medium* *Very Fast* *Fastest!*

Tracy command: Assignment

```
name = value
```



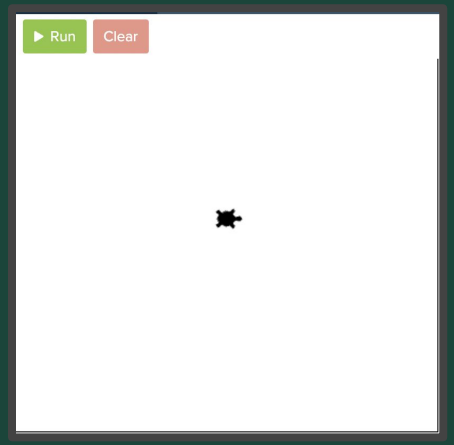

```
radius = 20
circle(radius)
```



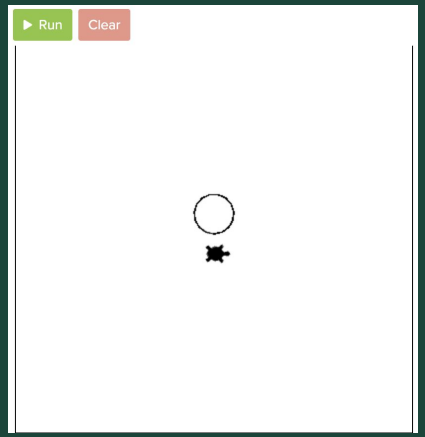
More About Variables

A **variable** is like a “box” for a “value” (number or text)

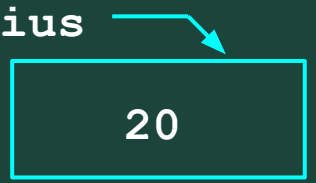
What's the Deal With Variables?



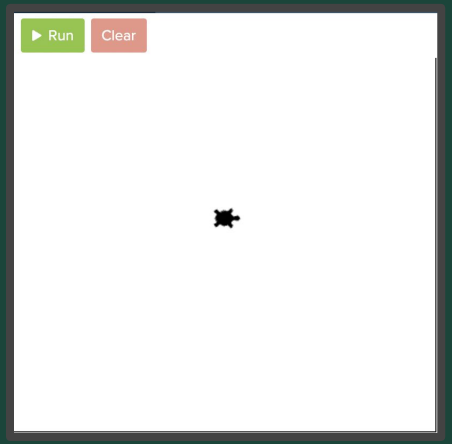

```
radius = 20
circle(radius)
penup()
setposition(0,-radius)
pendown()
```



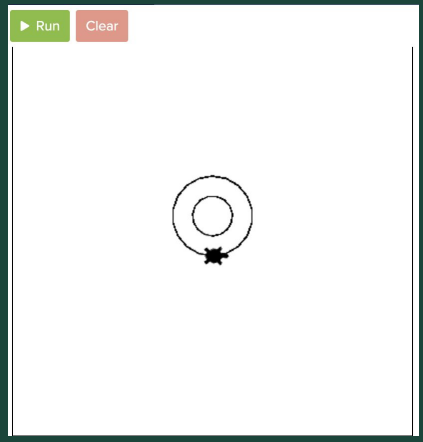
Variables let us set a value *once*, and then use that same value over and over!



What's the Deal With Variables?

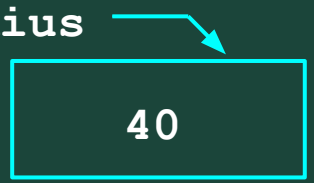



```
radius = 20
circle(radius)
penup()
setposition(0,-radius)
pendown()
radius = 2*radius
circle(radius)
```

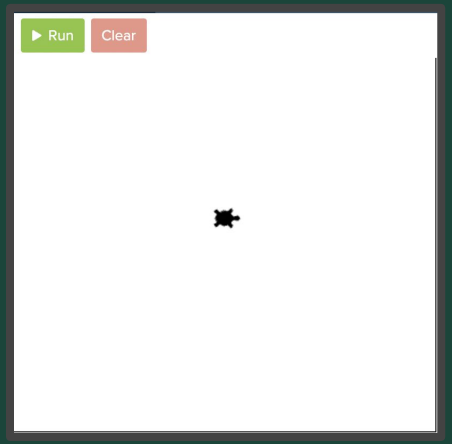


Variables let us set a value *once*, and then use that same value over and over!

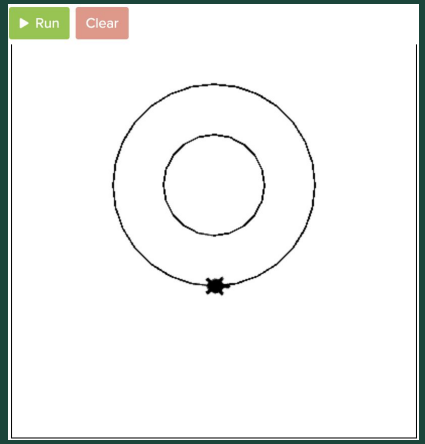
Variables can be modified to store different values over time!



What's the Deal With Variables?




```
radius = 50
circle(radius)
penup()
setposition(0,-radius)
pendown()
radius = 2*radius
circle(radius)
```



Variables let us set a value *once*, and then use that same value over and over!

Variables can be modified to store different values over time!

Variables make it easier to read and modify code!

Rules for Naming Variables

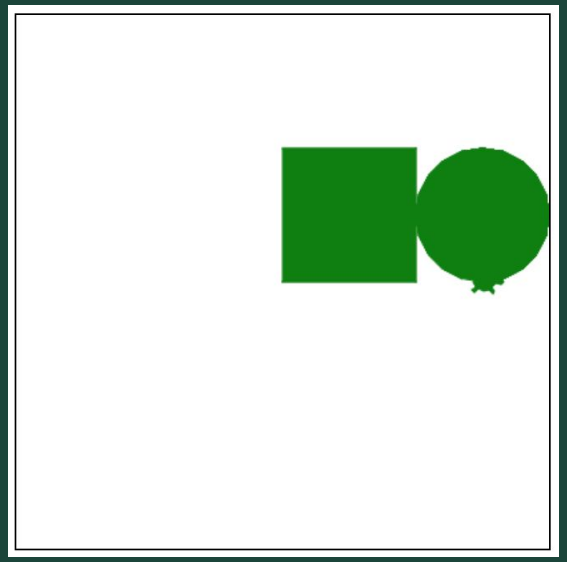
A V**ariable Name**:

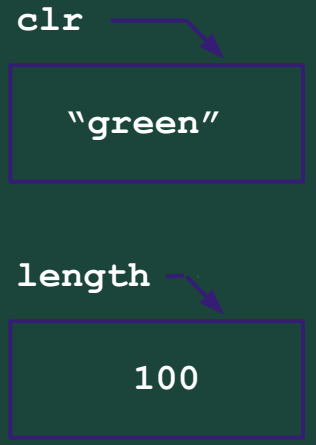
- Should start with a letter
- Consists of letters, numbers, and underscores (‘_’)
- Cannot be a Python “reserved word”
  (e.g., `int`, `float`, `input`, `print`, …)
- Should be meaningful – for example:

```
radius          speed          hair_color
```

Example

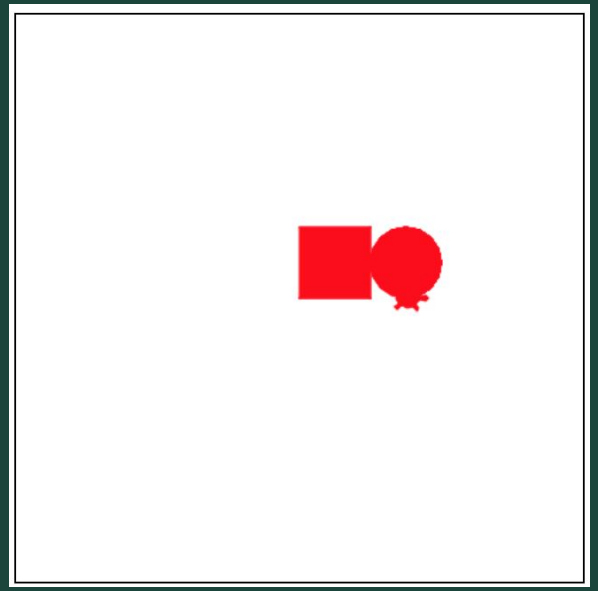
```
clr = "green"
length = 100

color(clr)
begin_fill()

forward(length)
left(90)
forward(length)
left(90)
forward(length)
left(90)
forward(length)
left(90)

end_fill()

penup()
forward(length + length/2)
pendown()

begin_fill()
circle(length/2)
end_fill()
```





Example

```
clr = "red"
length = 50

color(clr)
begin_fill()

forward(length)
left(90)
forward(length)
left(90)
forward(length)
left(90)
forward(length)
left(90)

end_fill()

penup()
forward(length + length/2)
pendown()

begin_fill()
circle(length/2)
end_fill()
```



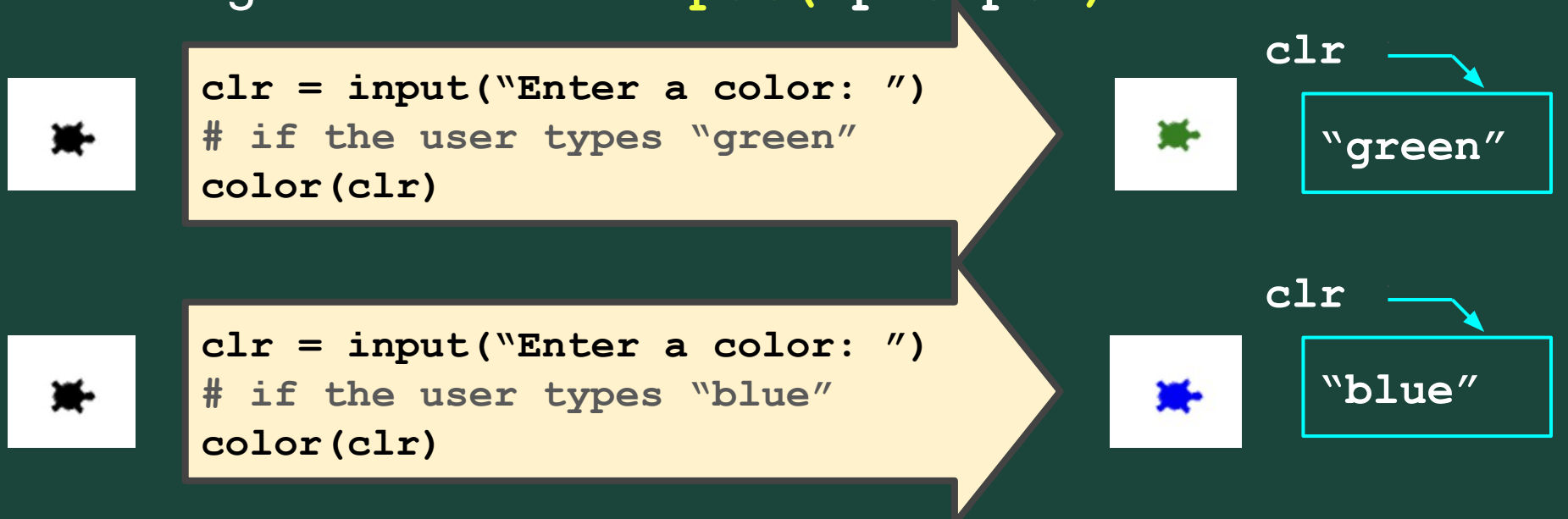
clr → "red"

length → 50

User Input (Text)

Variables can also be used to save *user input*, letting the user indicate values on-the-fly while running programs!

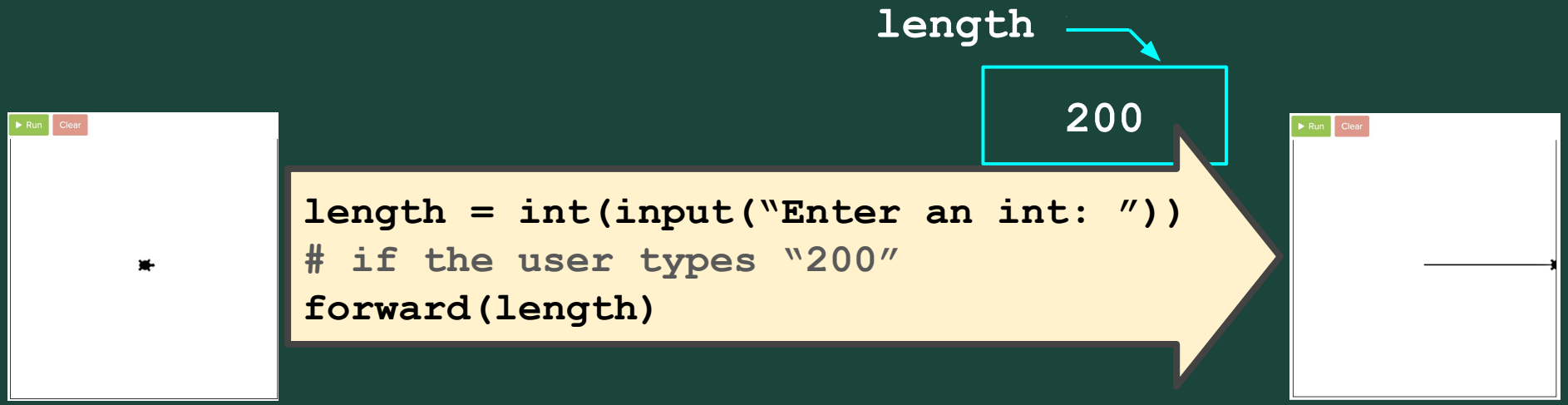
Use assignment with the `input("prompt")` function!



User Input (Numbers)

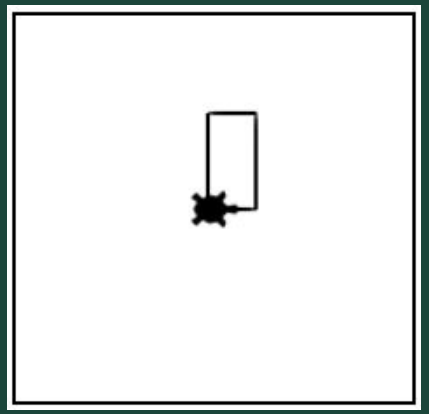
The user inputs a “string”, or sequence of characters.

To use a user input as a number, you have to tell Python to convert it to an “int” or a “float” – use `int(...)` or `float(...)`

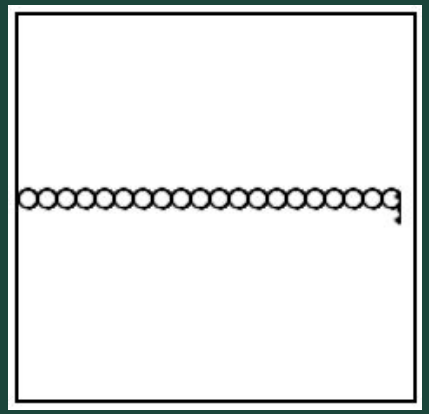


Coding Challenges

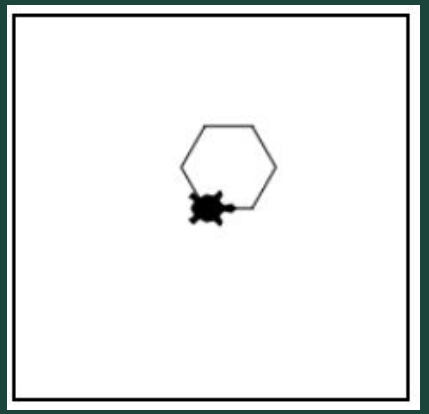
Challenge 1: Rectangle



Challenge 2: Circles



Challenge 3: Hexagon



Command	What does it do?
`color("color name")`	Changes Tracy's trail color
`pensize(number)`	Changes Tracy's trail thickness
`begin_fill()`	Starts tracking closed shapes
`end_fill()`	Fills & stops tracking closed shapes
`setposition(x, y)`	Moves Tracy to the input coordinates
`speed(number)`	Sets how fast Tracy executes commands
`name = value`	Saves the value in the variable
`input("prompt")`	Prints prompt and waits for user input
`int(...), float(...)`	Converts a value to a number (`int` or `float`)

Standup

- What is a challenge you faced today?
- What is something you were successful with?
- What do you want to improve on for next week?
- What are you most excited to learn?

Temperature Check

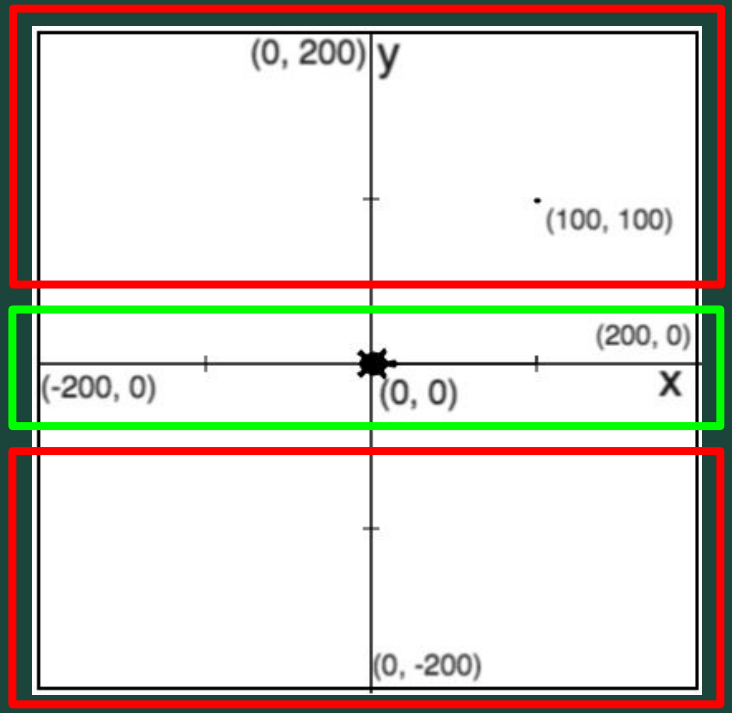
https://forms.gle/2RFvixVSe7C5vzzz5

Spotlight

Growth Mindset

- Why do you think it’s important to have a growth mindset?
- How can a growth mindset help you in Technovation?

Turning Tracy

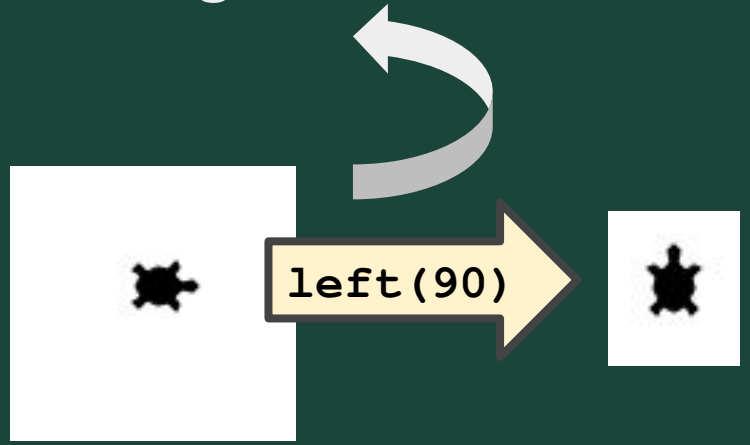


Tracy **always** starts facing right.

Tracy Command: left and right

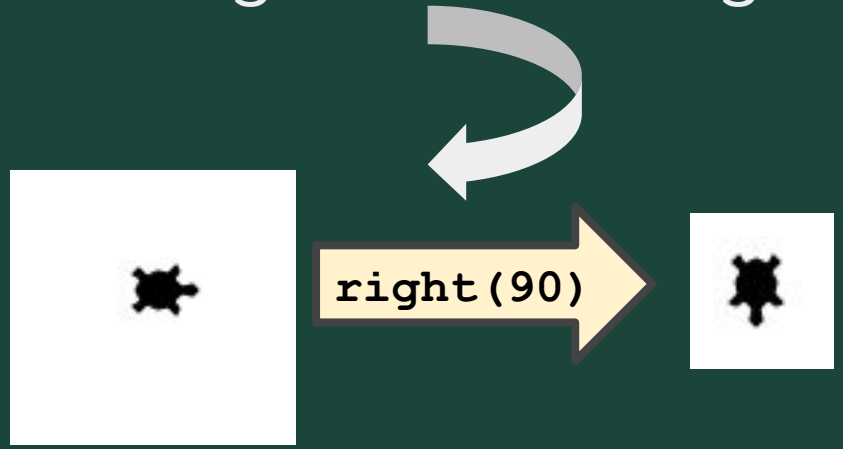
`left(90)`

Turns Tracy 90 degrees to the left



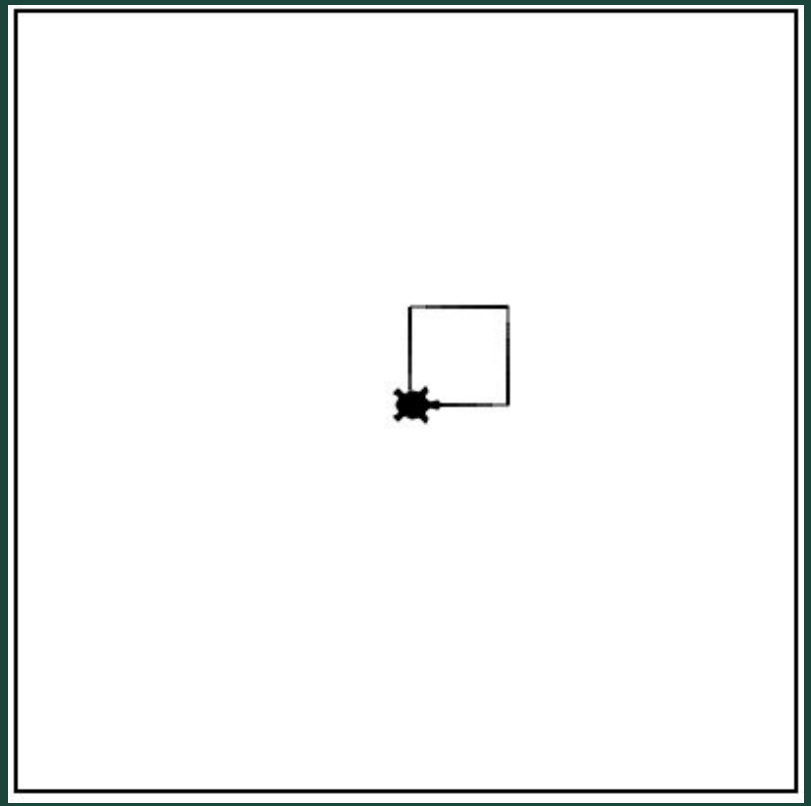
`right(90)`

Turns Tracy 90 degrees to the right



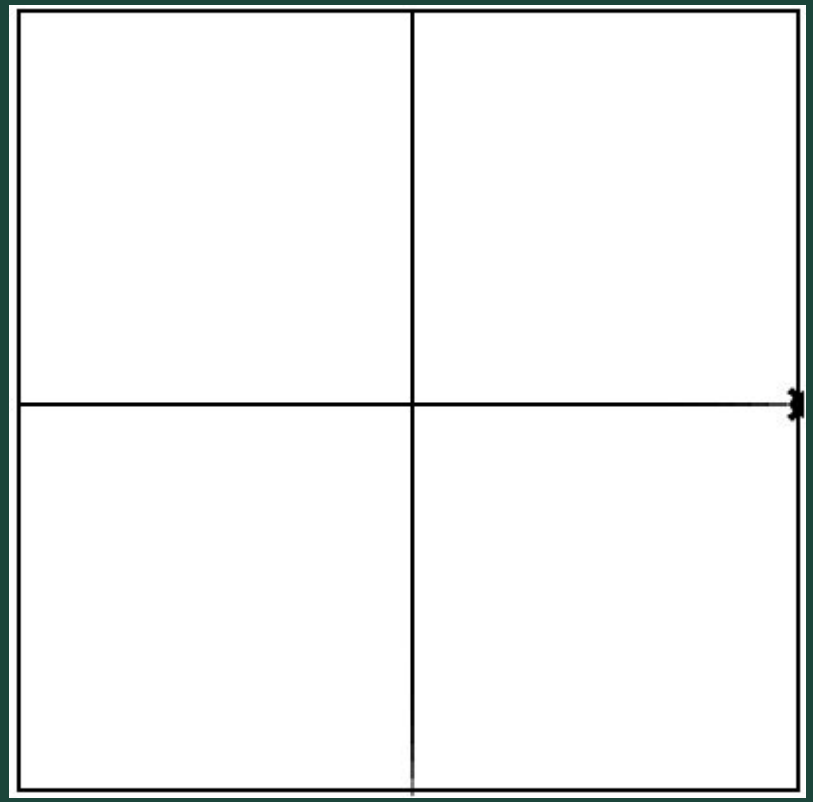
Example #1: Square

Write a program that has Tracy draw a square with sides of 50 pixels.



Example #2: X and Y axes

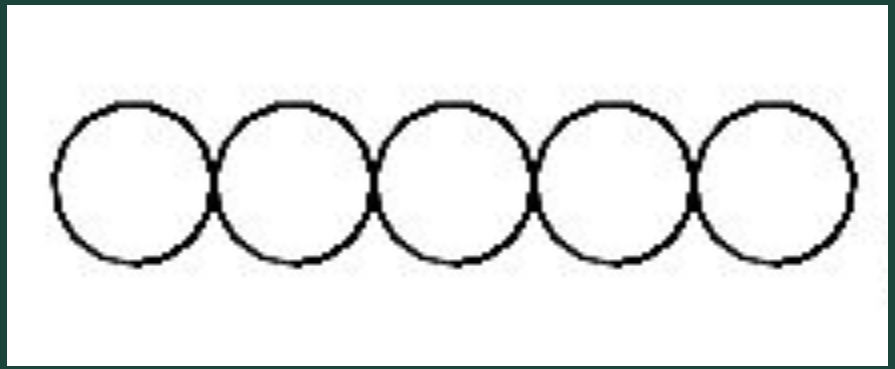
Write a program that has Tracy draw an x- and y-axis on the canvas.



Introducing Loops!

```
1  circle(20)
2  penup()
3  forward(40)
4  pendown()
5  circle(20)
6  penup()
7  forward(40)
8  pendown()
9  circle(20)
10 penup()
11 forward(40)
12 pendown()
13 circle(20)
14 penup()
15 forward(40)
16 pendown()
17 circle(20)
18 penup()
19 forward(40)
```

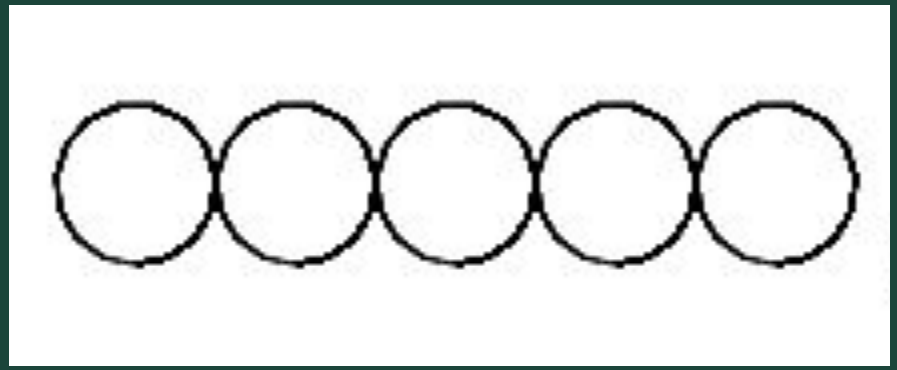
For loops are used to repeat code a fixed number of times.



Introducing Loops!

```
1  Tracy, repeat this code 5 times!
2      circle(20)
3      penup()
4      forward(40)
5      pendown()
```

For loops are used to repeat code a fixed number of times.



Introducing Loops!

```
1  circle(20)
2  penup()
3  forward(40)
4  pendown()
5  circle(20)
6  penup()
7  forward(40)
8  pendown()
9  circle(20)
10 penup()
11 forward(40)
12 pendown()
13 circle(20)
14 penup()
15 forward(40)
16 pendown()
17 circle(20)
18 penup()
19 forward(40)
```

For loops help us by:

- shortening our code
- making it easy to alter our code

19 lines to 5 lines!

```
1  Tracy, repeat this code 5 times!
2      circle(20)
3      penup()
4      forward(40)
5      pendown()
```

Introducing Loops!

```
1  circle(20)
2  penup()
3  forward(40)
4  pendown()
5  circle(20)
6  penup()
7  forward(40)
8  pendown()
9  circle(20)
10 penup()
11 forward(40)
12 pendown()
13 circle(20)
14 penup()
15 forward(40)
16 pendown()
17 circle(20)
18 penup()
19 forward(40)
```

For loops help us by:

- shortening our code
- making it easy to alter our code

Change radius to 50 pixels

```
1  Tracy, repeat this code 5 times!
2      circle(20)
3      penup()
4      forward(40)
5      pendown()
```

Writing For Loops

```
for i in range (amount of times to repeat):
    Commands to repeat go here (indented!)
```

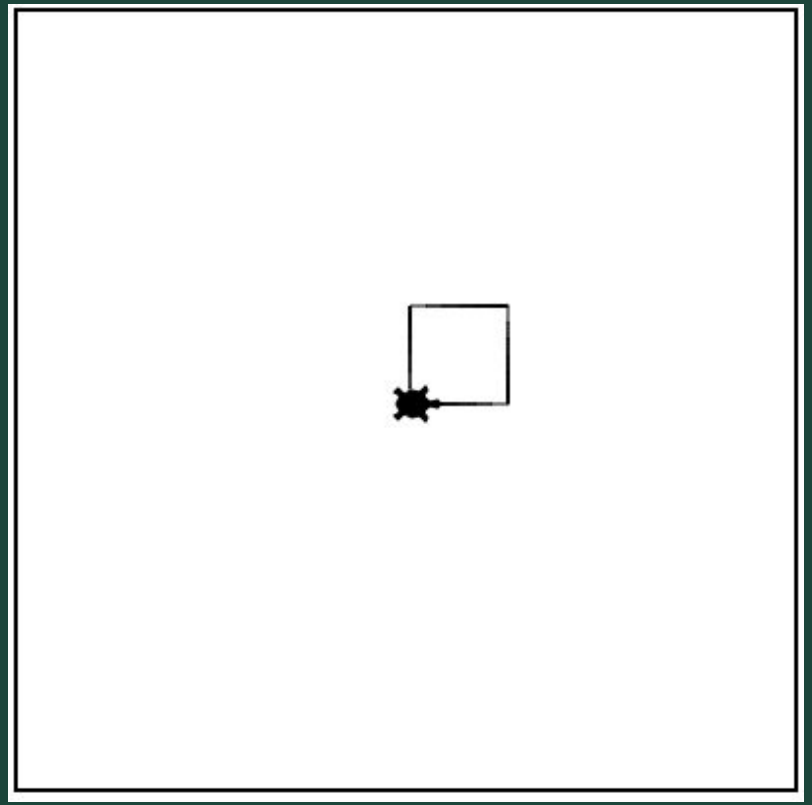
```
1  Tracy, repeat this code 5 times!
2      circle(20)
3      penup()
4      forward(40)
5      pendown()
```

Write loop

```
1  for i in range(5):
2      circle(20)
3      penup()
4      forward(40)
5      pendown()
```

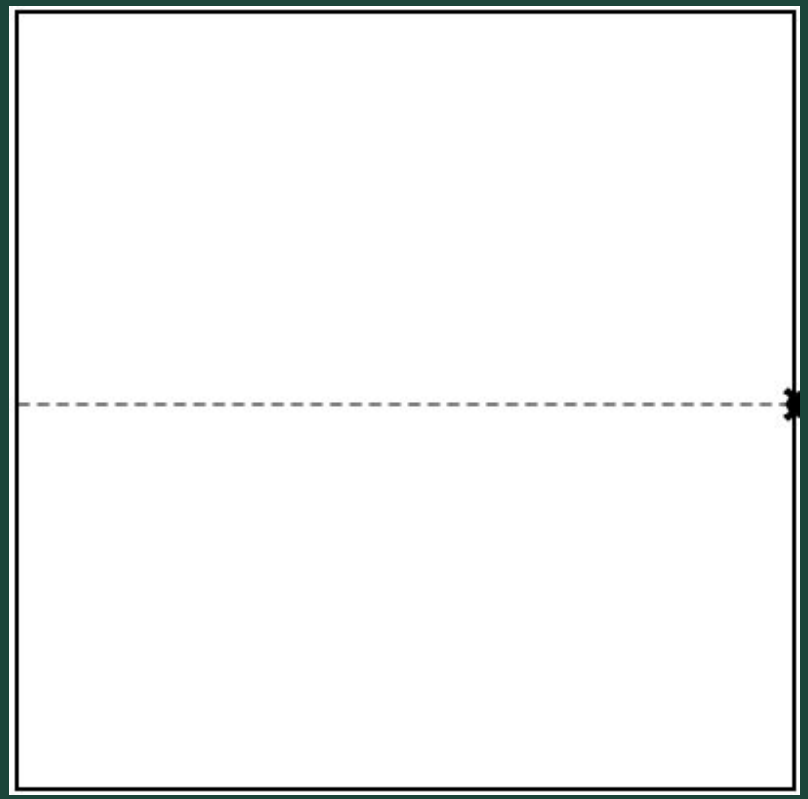
Example #3: Square using for loops

Write a program that has Tracy draw a square with sides of 50 pixels using a for loop.



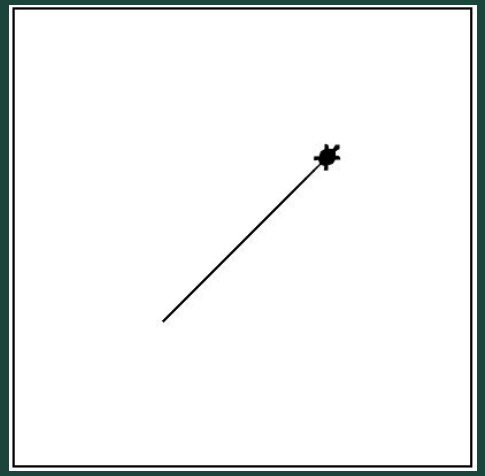
Example #4: Dotted Line

Write a program that has Tracy draw a dotted line along the x-axis of the canvas.



Turning Tracy - Limitations

What if we want Tracy to draw diagonal lines?

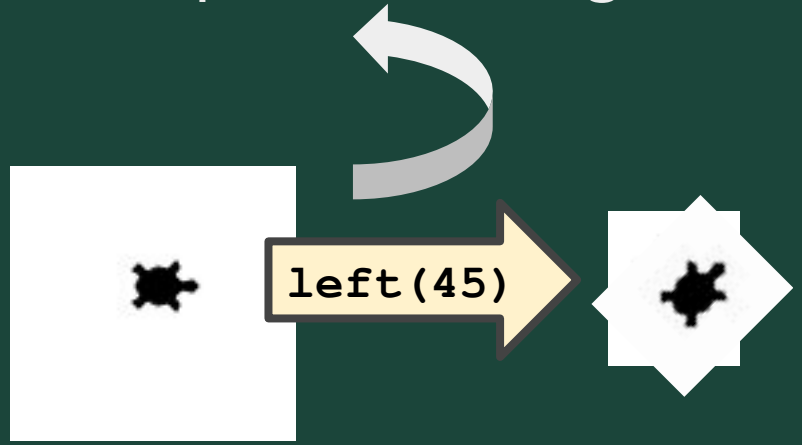


Right now, Tracy can only turn at 90 degree angles but we need her to be able to turn at any angle we wish.

Advanced Turning Commands

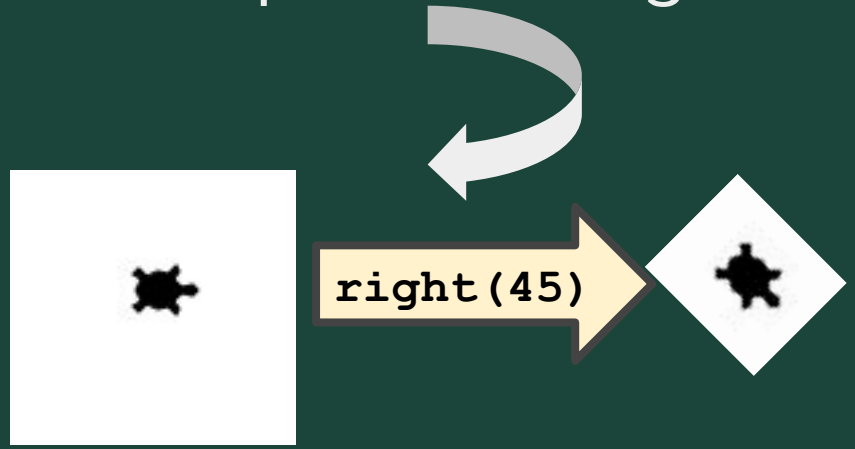
`left(angle)`

Turns Tracy left at a specified angle



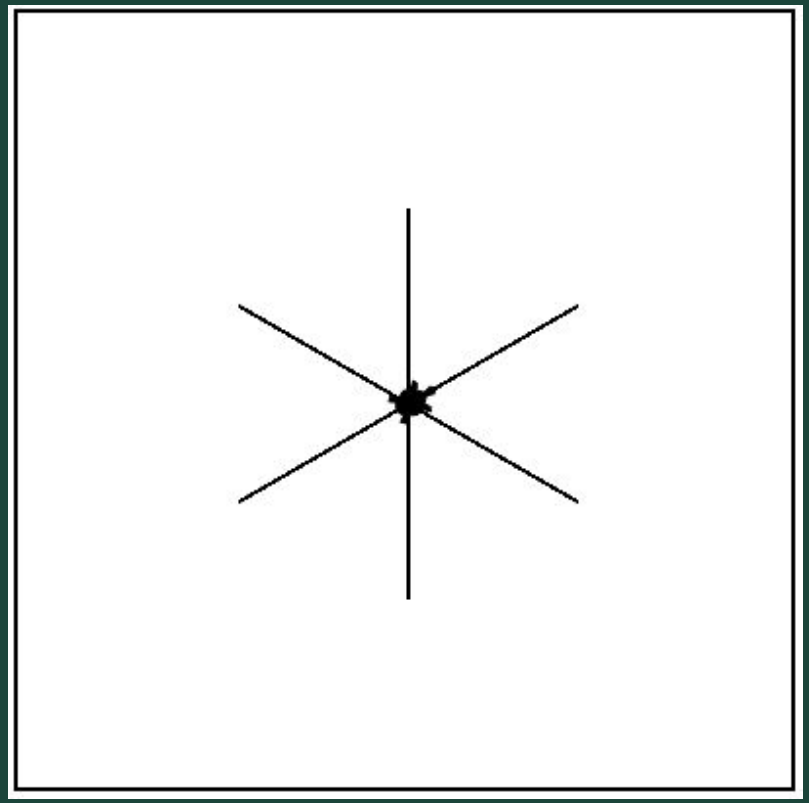
`right(angle)`

Turns Tracy right at a specified angle



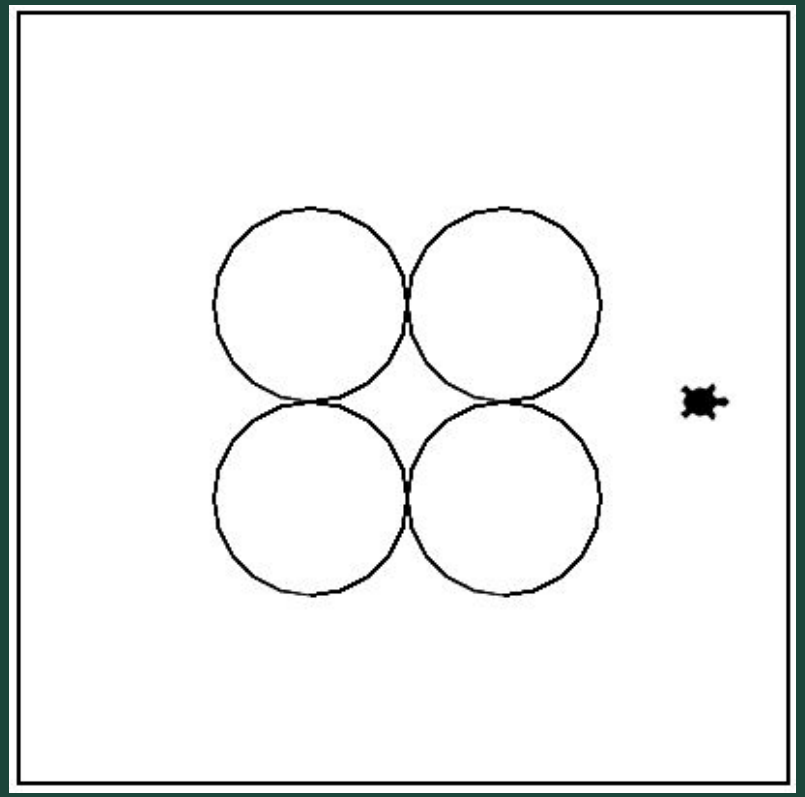
Example #5: Asterisk

Write a program that has Tracy draw an asterisk with lines 100 pixels long.



Example #5: Four Circles

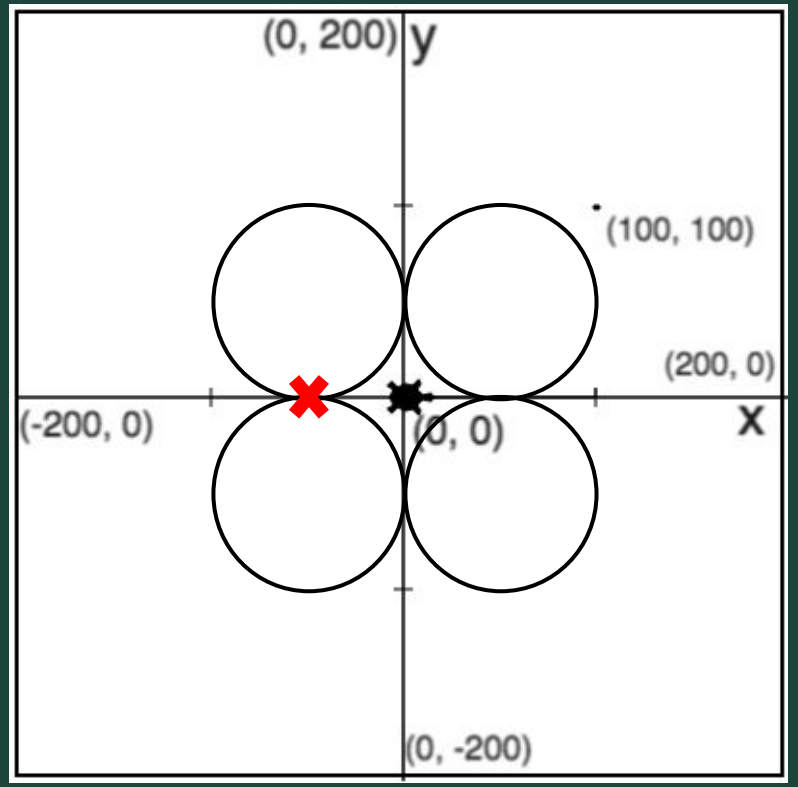
Write a program that has Tracy draw four circles at the center of the canvas.



Example #5: Four Circles

Notes:

- 2 x radius = diameter
- Bottom row starting position: (-50,-100)
- Distance between circles: 100
- Top row starting position: (-50, 0)



More challenges!

- See if you can draw the following shapes in Sandbox!