Upperline Code
Functions

What Is A Function?

A function is a group of instructions that performs a certain task.

Example:

`CIRCLE_UP()`	`DRAW_SQUARE()`

• Stand up.
• Push in chair.
• Turn toward the circle area.
• While not at the circle area:
  ○ Move forward.
• Face the teacher.
	• Repeat 4 times:
	  ○ Move forward 20 pixels.
	  ○ Turn right 90 degrees.

How To Use Functions?

1) Define the Function – These blocks of code lay out each of the steps – one line of code at a time - the overall action the function is supposed to perform.

2) Call the Function – Write the function name to execute the code inside the function.

Why Use Functions?

You can define your own functions to do what you want!

```
def function_name():
    # This code is inside the function
    # You can program your function to
    #   do whatever you want!

# This code is outside the function

# Call your function like this:
function_name()
```

Functions In Python

You've (probably, hopefully) already seen Python functions!

- print() #used to display statements
- range() #used typically in for loops

These functions are built-in inside the Python programming language

Practice: Define And Call Your Own Functions

- Take a look at the code. Can you guess what will happen when you run it?
- Run it via the command line. What does it do?

```
# --- Define your functions below! ---



# --- Put your main program below! ---
def main():
  while True:
    answer = input("(What will you say?) ")
    print("That's cool!")



# DON'T TOUCH! Setup code that runs your main() function.
if __name__ == "__main__":
  main()
```

Practice: Define And Call Your Own Functions

Write a function so that the program introduces itself to the user before it starts asking for input.

```
# --- Define your functions below! ---


# --- Put your main program below! ---
def main():
  while True:
    answer = input("(What will you say?) ")
    print("That's cool!")


# DON'T TOUCH! Setup code that runs your main() function.
if __name__ == "__main__":
  main()
```

What is Scope?

- Scope is the part of a program where a particular variable is visible.

	GLOBAL SCOPE	LOCAL SCOPE
Set	main body of a file	inside a function
Visible	everywhere within a file!	only inside the function.

Scope - Examples

- The variable, name: is it a global variable or a local variable? Why?

- What will happen when this code runs?

```
def say_hello():
    print("Hi " + name)

name = "Michelle"
say_hello()
```

Global Variables

- Global variables are considered bad programming practice!
  - Hard to tell which function(s) are reading/writing to a variable
  - Harder to reuse code across programs

- Avoid using global variables whenever possible!!

Scope - Examples

- The variable, name: is it a global variable or a local variable? Why?
- What will happen when this code runs?

```
def say_hello():
  print("Hi " + name)

def main():
  name = "Michelle"
  say_hello()

if __name__ == "__main__":
  main()
```

What Do We Do?

Global variables are bad practice.

Local variables give NameErrors.

Next: Introduce Parameters…