# Intro to programming

## Syntax basics with examples in Python

# What is programming?

# What is PyLadiesATL?

## PyLadies

> We are an international mentorship group with a focus on helping more women become active participants and leaders in the Python open-source community.

## PyLadiesATL

- **Monthly Meetup**
  - 4th Thursday night of the month
  - Location: DigitalCrafts (A.T.V.), 3423 Piedmont Rd NE
- **All the Nerdy Ladies social hour** (co-sponsored)
  - 1st Monday night of the month
  - Joystick Gamebar, 427 Edgewood Ave SE

### Poetry

Words are used for their sounds as well as for their meaning, and the whole poem together creates an effect or emotional response. Ambiguity is not only common but often deliberate.

### Prose

The literal meaning of words is more important, and the structure contributes more meaning. Prose is more amenable to analysis than poetry but still often ambiguous.

### Programs

The meaning of a computer program is unambiguous and literal, and can be understood entirely by analysis of the tokens and structure.

> How to Think Like a Computer Scientist, Chapter 1

# Tips for reading programs

### It takes longer to read them

Formal languages (programs) are much more dense than natural languages. Read carefully.

### Structure is important

The structure of a program is not always as simple as a top-to-bottom flow, something to keep in mind when navigating code.

### Details are extremely important

Typos, incorrect punctuation, unclosed parentheses or quotes -- tiny details can cause big problems.

# Start your REPLs

1. **R**ead some code written by the user
2. **E**valuate the code as computer instructions
3. **P**rint any output from interpreting the code
4. **L**oop (continue repeating the first 3 steps)

## repl.it/languages/python3

```
print("Hello World!")
```

# Programming syntax

## Variables

Names assigned to pieces of data

```
fav_number = 172
print(fav_number + 3)
```

## Math operators

Yes, we can compute!

```
add_answer = 9 + 91
sub_answer = 10.0 - 3
mult_answer = 1.1 * 7.0
div_answer = 1 / 5
```

# Programming syntax

## Functions

Functions are reusable units of code. They encapsulate a set of instructions that, once defined, can be called again and again from other parts of the code.

```
def seven_times(z):
  """ This function returns 7 times whatever z is. """
  return 7 * z


print(seven_times(7))
print(seven_times(0.5))
print(seven_times(1111))
print(seven_times(.0001))
```

# Programming syntax

## Types

Specifically, of what *type* is a given piece of data?

```
my_string = "hello"
my_integer = 17
my_float = 3.0
my_boolean = True

""" builtin type() method returns type info: """
print(type(8.0))
```

## Type casting (conversion)

```
print(type(str(7)))
print(type(7))
```

# Examples

- Example 1: bouncer
- Example 2: donuts
- Example 3: going out?
- Example 4: replace instances of first letter
- Example 5: mixup
- Example 6: both ends

# What next?

- Google's Python Class
- Python documentation
- Automate the Boring Stuff with Python
- Python for Neuroscience
- PyLadies
- PyLadiesATL

# The end