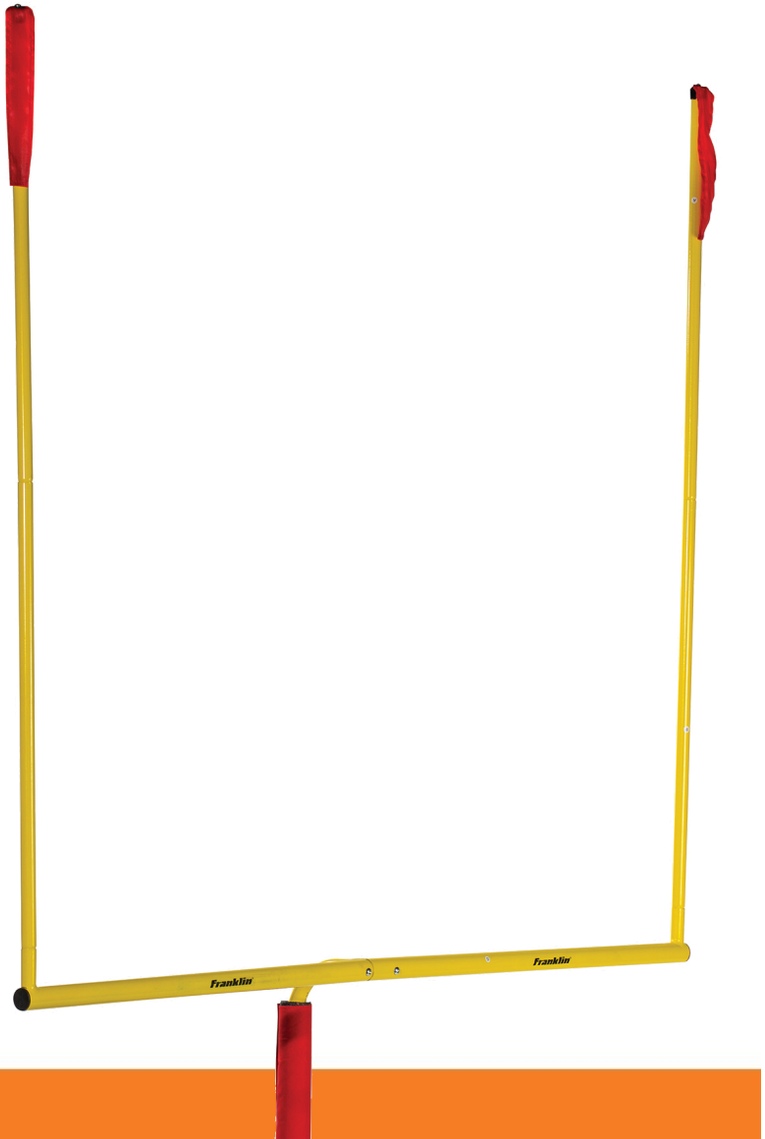
Primitives

Develop Intelligence

Goals

1. Explain `None`
2. List 3 `falsy` values
3. Describe how to use a **format string**



Roadmap

1. Types of Types
2. Numbers
3. Booleans
4. Strings

Develop
Intelligence

Types of Types

Develop Intelligence

What's a 'Type'?

- Shape of data
- e.g.
  - Letter
  - Number
  - List
  - Table

Basic Types

- **Atoms**
  - Indivisible
  - e.g. String, Number, Boolean, `None`
- **Containers**
  - Hold atoms, objects, or more containers
  - e.g. List, Dictionary, Tuple
- ***Everything*** is an object
  - i.e. Python doesn't really have primitives

Get Type information

- Useful built-ins:
  - `type`
  - `help`
  - `dir`

```
# Get the type
type(12)
type('chicken')

# Help
x = 1.2
help(x)
```

The special case of `None`

- Represents the absence of a value
- Equivalent to `null` in C++,Java,etc
- Implicit function return value

```
x = None
y = print(x)
if y == None:
  print('Y is none too!')
```

Develop
Intelligence

Numbers

Develop Intelligence

Overview

- Can be integers or reals
- Limited precision
- Arbitrarily big

```
1 google = 10**100
2 googolplex = 10 ** 10 ** 100
3
4 pi = 3.14159265358979323846264338327950288419716939937510582097494459230
```

Easy Integers

- Operators work as expected

```
x, y = 3, 2

print(x + y) # = 5
print(x - y) # = 1
print(x * y) # = 6
print(x / y) # = 1.5
print(x // y) # = 1 (Integer division)
print(x % y) # = 1 (Modulo)
print(int(3.9)) # = 3
print(float(x)) # = 3.0
print(x ** y) # = 9 (Exponent operator)
```

Converting Integers

- Use the 'constructor' function `int`

```
1 import time
2
3 time_text = input('Enter the cooking time in seconds: ')
4 time_int = int(time_text)
5 print('Put the egg in boiling water now')
6
7 time.sleep(time_int)
8 print('Take the egg out now')
```

Converting Floats

- Use the `float` constructor

```
time_text=input('Enter the cooking time in seconds: ')
time_float=float(time_text)
sleep(time_float)
```

Comparison Operators

```
# Equals
if 1 == 1:
    print('bored')

# Not equals
if 1 != 0:
    print('bored')

# Greater than
if 1 > 0:
  print('bored')

# Greater than or equals
if 1 >= 0:
  print('bored')
```

Develop
Intelligence

Booleans

Overview

- `True` and `False`
- Constructor: `bool`

```
1 x = 1 > 2
2 print(x) # False
3 y = 2 > 1
4 print(y) # True
```

Turn Anything Into Boolean

```
colors=['red','red','green','red']
colors_as_boolean = bool(colors)
print(colors_as_boolean)

pi = 3.14
pi_as_boolean = bool(pi)
print(pi_as_boolean)

label = 'False'
label_as_boolean = bool(label)
print(label_as_boolean)
```

Only Falsy Things

```
1  empty_list = []
2  zero = 0
3  false=False
4  empty_string=''
5  none=None
6
7  if bool(empty_list) or bool(zero):
8    print("Truthy!")
9  else:
10   print("Falsy")
```

Boolean Operators

```
x, y = True, False

# Or
print(x or y) # True

# And
print(x and y) # False

# Not
print(not y) # True
```

Develop
Intelligence

Strings

Develop Intelligence

Overview

- Constructor: `str`
- Delimited with ' or "
- Immutable
- Unicode support

```
1 message1 = "I've got an apostraphe."
2 message2 = 'Dude. "Big Changes" are coming."
3 message3 = "I'm a 🦙 and " + message1
```

Format Strings

- For interpolation prefix with ***f***
- Put information in brackets ***{}***

```
id=11
name = 'Sandy'
print(f"Hello {name} your id is: {id}")
```

Stupid `str` tricks

Develop Intelligence

```
1 'Dr. Who'.removeprefix('Dr. ')
2 '1,2,3'.split(',')
3 'Hello world'.title()
```

Lab: Monty Hall (Optional)

Game plan

- In VSCode, open the directory $/python_foundations/labs/
- Read instructions: src/Q.O.MontyHall/README.md
- Build the applications starting with 'startingpoint'

Develop
Intelligence

Review

1. Explain `None`
2. List 3 `falsy` values
3. Describe how to use a **format string**