Let's Learn Python!

Getting started:

1. Join the wifi network
   Network: CIC
   Password:

2. Get Python installed

3. Check Python version (python -V in the command line)

4. Start the Python interactive shell

5. Get ready to have fun!

What to expect:

- Programming fundamentals
- Lessons in Python
- Working with neighbors
- Example code

What is programming?

★ A **computer** is a machine that **stores** and **manipulates** information

★ A **program** is a detailed set of **instructions** telling a computer exactly what to do.

Instructions:

People vs. Computers

Algorithms

97 Simple Steps to a PB&J

Is making PB&J difficult?

How many steps does it feel like?

Let's talk to Python!

Why Python?

- ★ Readable syntax
- ★ Powerful
- ★ Awesome community
- ★ Interactive shell

Python Interactive Shell

aka, the Python interpreter

- ★ Easy to use
- ★ Instant feedback

The **prompt** - prompts you to type stuff:

```
>>>
```

Numbers

Numbers

Arithmetic operators:

addition: +

subtraction: -

multiplication: *

division: /

Try doing some math in the interpreter:

```
>>> 1 + 2
>>> 10 / 3
```

Is the last result what you expected?

Numbers

Integers (whole numbers):

```
9
-55
```

Floats (decimals):

```
17.318
10.0
```

```
>>> 11/3
3
```

```
>>> 11.0/3.0
3.6666666666666665
```

Rule:

★ If you want Python to respond in floats, you must talk to it in floats.

Strings

Strings

```
>>> “abcdef”
>>> “garlic breath”
>>> “Thanks for coming!”
```

Try typing one without quotes:

```
>>> apple
```

What’s the result?

Strings

If it's a string, it must be in quotes.

```
>>> "apple"
>>> "What's for lunch?"
>>> "3 + 5"
```

Rules:

- ★ A string is a character, or sequence of characters (like words and sentences).
- ★ A number can be a string if it's wrapped in quotes

Strings

String operators:

concatenation (joining words together): +

multiplication: *

Try concatenating:

```
>>> “Hi” + “there!”
‘Hithere!’
```

Try multiplying:

```
>>> “HAHA” * 250
```

Strings: Indexes

Strings are made up of **characters**:

```
>>> “H” + “e” + “l” + “l” + “o”
‘Hello’
```

Each character has a position called an *index*:

```
H e l l o
0 1 2 3 4
```

In Python, indexes start at **0**

Strings: Indexes

```
>>> print “Hey, Bob!”[4]
```

What did Python print?

Rules:

- ★ Each character’s position is called its *index*.
- ★ Indexes start at 0.
- ★ Spaces inside the string are counted.

The print command

print

Without print, you can concatenate with the '+' operator:

```
>>> "This" + "isn't" + "great."
Thisisn'tgreat.
```

With print, you get spaces between items:

```
>>> print "This", "is", "awesome!"
This is awesome!
```

Practicing with print

```
>>> print “Jennifer has”, 2,
“cats.”
Jennifer has 2 cats.
```

```
>>> print 6+6, “eggs make a dozen.”
12 eggs make a dozen.
```

Try printing two sentences with numbers outside the quotes.

Variables

Variables

Calculate a value:

```
>>> 12 * 12
144
```

How can you save that value, 144?

Assign a name to a value:

```
>>> donuts = 12 * 12
>>> color = “yellow”
```

A **variable** is a way to *store a value*.

Variables

- ★ Calculate once, keep the result to use later
- ★ Keep the name, change the value

Some other things we can do with variables:

```
>>> fruit = “watermelon”
>>> print fruit[2]
>>> number = 3
>>> print fruit[number-2]
```

Variables

Converting variables:

Turn a string into a number (use int or float).
(Notice that pets is really a string.)

```
>>> pets = '4'
>>> num_pets = int(pets)
```

Turn a number into a string:

```
>>> pets_2 = str(num_pets)
```

Variables Practice

```
>>> name = “Your Name”
>>> color = “Your Favorite Color”
>>> print “My name is”, name, “and my favorite color is”, color
```

What’s the output? Is it what you expect?

Errors

Errors

```
>>> “friend” * 5
‘friendfriendfriendfriendfriend’

>>> “friend” + 5
Error

Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: cannot concatenate 'str' and 'int' objects
```

Do you remember what ‘concatenate’ means?

What do you think ‘str’ and ‘int’ mean?

Errors

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: cannot concatenate 'str' and 'int' objects
    ^                        ^           ^          ^
```

- Strings: ‘str’
- Integers: ‘int’
- Both are objects
- Python cannot concatenate objects of different *type*s

Errors

Here's how we would fix that error:

```
>>> "friend" + 5
Error
```

Concatenation won't work.

Let's use the `print` command for display:

```
>>> print "friend", 5
friend 5
```

No concatenation, no problem!

Exercise Set 1

1. Store your name, height and favorite color in variables. Print that information in a sentence.

2. Calculate the number of 2-week disposable contact packs you need in a year and store that value in a variable. Print, in sentence form, the number of disposable contact packs you need to buy to be stocked for two years.

3. Calculate how many seconds all attendees will spend in this meetup and store it in a variable. Print the answer as a sentence.

4. Calculate the number of donuts made in a week if 15 dozen are made each day. Print, in sentence form, the number of donuts 100 people would have to eat in order to finish them all.

Exercise Set 1: Review

Store your name, height and favorite color in variables. Print that information in a sentence.

```
>>> name = “Jennifer”
>>> height = “65”
>>> color = “blue”
>>> print name, “is”, height,
    “inches tall and loves”, color, “!”

Jennifer is 65 inches tall and loves
blue!
```

Exercise Set 1: Review

Calculate the number of 2-week disposable contact packs you need in a year and store that value in a variable.

```
>>> contact_packs = 52 / 2
```

Print out, in sentence form, the number of disposable contact packs you need to buy to be stocked for two years.

```
>>> print “I will need to buy”, contact_packs,
“contact packs this year.”

I will need to buy 26 contact packs this year.
```

Exercise Set 1: Review

Calculate how many seconds all attendees will spend in this meetup.

Store that value in a variable.

```
>>> seconds = 60 * 60 * 2 * 30
```

Print the answer in a sentence.

```
>>> print “Attendees will spend a total
of”, seconds, “seconds in this meetup.”

Attendees will spend a total of 216000
seconds in this meetup.
```

Exercise Set 1: Review

Calculate the number of donuts made in a week if 15 dozen are made each day.

```
>>> number_of_donuts = 15 * 12 * 7
```

Print, in sentence form, the number of donuts 100 people would have to eat in order to finish them all.

```
>>> print “Each person will eat”,
number_of_donuts / 100.0, “donuts.”

Each person will eat 12.6 donuts.
```