The workshop will begin at 10:02 Central

Python Fundamentals
Tuesday: Lists, Loops, and Ifs

Colby Witherup Wood

NU IT Research Computing Services

About this bootcamp

Cameras are **not** required during these lectures.

During lectures, ask questions in the Zoom chat. If you know the answers, feel free to answer questions in the chat, too.

If my internet goes out during a lecture, that means everyone gets a 10-minute break! (Has never happened, but it's good to have a plan.)

Objects and functions

The two main concepts of Python.

Object: a particular piece of data (like a noun)

Function: something you can do to/with an object (like a verb)

Objects and functions

You can use certain functions with certain classes of objects.

Some functions are shared between classes of objects, and some are not.

Objects and functions

Yesterday's homework included string functions.

All functions have limitations and some work in interesting ways.

For example:

```
"I'll be there".title()
```

```
"I'Ll Be There"
```

Objects and functions

Once you've completed Wednesday's work, you'll be more equipped to create your own workarounds for these limitations.

Try not to get too hung up on the limitations! You can always use logic to code a workaround.

Objects and functions

If your research requires you to frequently search for patterns in strings, there is a package called **re** (regular expressions) that gives you a lot more control over string searches and manipulations.

I will be covering some basics of this package in one of my Next Steps in Python Lunch Lessons in 2021.

Plan for the week – **6 basic objects**

LEARN PYTHON

Monday

- Objects
  - **integers**
  - **floats**
  - **booleans**
  - **strings**
- Functions
  - basic operators
  - convert between objects
  - string functions
- Concepts
  - naming variables
  - indexing strings

Tuesday

- Objects
  - **lists**
- Functions
  - list functions
- Concepts
  - for loops
  - if statements
  - error handling
  - importing functions from modules

Wednesday

- Objects
  - **dictionaries**
  - files
- Concepts
  - writing files
  - looping through dictionaries
  - writing your own functions

Thursday

- Concepts
  - interactive coding
  - writing scripts
  - running scripts
  - putting it all together

DATAFRAMES

Friday

- Objects
  - pandas dataframes
  - pandas series
  - plots
- Functions
  - dataframe functions
  - series functions
- Concepts
  - selecting data
  - filtering data
  - plotting data

Options to open tuesdayLecture.ipynb:

If you are running Jupyter Lab on your own computer:

- Go to https://github.com/aGitHasNoName/pythonBootcampTuesday. Click on the green Clone button and choose Download ZIP. Unzip that folder. Open Anaconda Navigator and open Jupyter Lab. Navigate to the folder you just unzipped. Open tuesdayLecture.ipynb

If you are using Google Colab (online)

- Go to colab.research.google.com. Choose GitHub on the orange menu. Search for agithasnoname/pythonbootcamptuesday. Choose tuesdayLecture.ipynb

Prepare your desktop setup

- You can set my Zoom share up next to your own notebook
- You can run the code in your version of the notebook while just listening to me as I walk through it
- You can just sit back and watch me run through it until it is time to do an exercise

Let’s code!

Try both. What is the difference?

```
frodo = ["hobbit"]
sam = ["hobbit"]
frodo
sam
frodo.append("ring")
frodo
sam
```

```
harry = ["wizard"]
ron = harry
harry
ron
harry.append("scar")
harry
ron
```

Try this.

```
frodo = ["hobbit"]
sam = frodo.copy()
frodo
sam
frodo.append("ring")
frodo
sam
```

Loops

```
for person in row:
    person.clap()
```

Loops

```
for person in row:
    person.clap()
```

For the next few examples, try to predict what is going to happen. Trust your gut.

```
for person in row:
    person.speak("Hello")
```

```
for person in row:
    person.speak("Hello")
```

```
for person in row:
    person.speak("Hello")
```

```
for person in row:
    person.speak("Hello")
```

```
for person in row:
    person.speak("Hello")
```

```
for person in row:
    person.speak("Hello")
```

```
for person in row:
    if person is Avenger:
        person.speak("Hello")
    else:
        person.speak("Sorry")
```

1

```
for person in row:
    if person is Avenger:
        person.speak("Hello")
    else:
        person.speak("Sorry")
```

1

Hello

```
for person in row:
    if person is Avenger:
        person.speak("Hello")
    else:
        person.speak("Sorry")
```

1

```
for person in row:
    if person is Avenger:
        person.speak("Hello")
    else:
        person.speak("Sorry")
```

1

```
for person in row:
    if person is Avenger:
        person.speak("Hello")
    else:
        person.speak("Sorry")
```

1

```
for person in row:
    if person is Avenger:
        person.speak("Hello")
    else:
        person.speak("Sorry")
```

1

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    else:
        pass
```

2

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    else:
        pass
```

2

Hello

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    else:
        pass
```

2

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    else:
        pass
```

2

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    else:
        pass
```

2

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    else:
        pass
```

2

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

3

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

Hello

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

Luke!

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

3

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

3

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

3

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        pass
```

4

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        pass
```

4

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        pass
```

4

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        pass
```

4

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        pass
```

4

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        pass
```

4

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        break
```

5

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        break
```

5

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        break
```

loop stops

for loop syntax

```
for p in row:
    p.speak("Hello")
```

one tab or four spaces – you pick, but it must match throughout an entire script

for loop syntax

```
for p in row:
    p.speak("Hello")
```

Temporary variable that you create for the for loop. Must match exactly.

if statement syntax

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
```

one tab or four spaces – you pick, but it must match throughout an entire script

if statement syntax

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
```

if statement syntax

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
```

We say that everything that is indented at least one tab under the for loop statement is **inside** the for loop.

if statement syntax

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
```

We say that everything that is indented at least one tab under the if statement is **inside** the if statement.

if statement syntax

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

Let’s code!

Checking for errors

Find the error on the next few slides.

```
for p in row:
    if p is Avenger and p can fly:
    p.speak("Hello")
    else:
    pass
```

```
for p in row:
    if p is Avenger and p can fly:
        p.speak("Hello")
    else:
        pass
```

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif person in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

```
for p in row:
    if p is Avenger:
        p.speak("Hello")
    elif p in Star Wars:
        p.speak("Luke!")
    else:
        pass
```

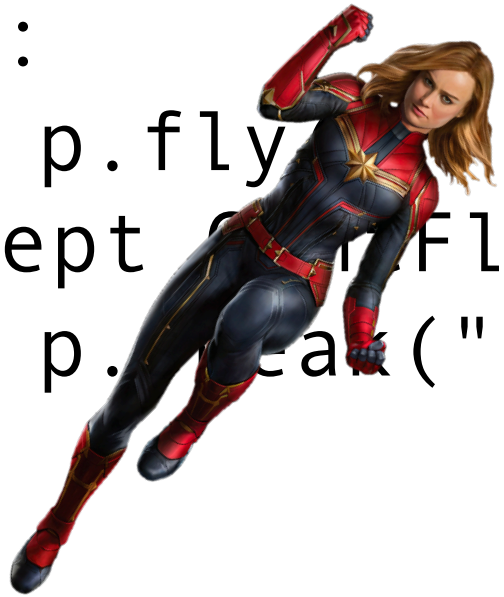
```
for p in row:
    p.fly()
```

```
for p in row:
    p.fly()
```

```
for p in row:
    p.fly()
```

```
for p in row:
    try:
        p.fly()
    except CantFlyError:
        p.speak("Sorry")
```

```
for p in row:
    try:
        p.fly
    except CantFlyError:
        p.speak("Sorry")
```



```
for p in row:
    try:
        p.fly()
    except CantFlyError:
        p.speak("Sorry")
```

```
for p in row:
    try:
        p.fly()
    except CantFlyError:
        p.speak("Sorry")
```

```
for p in row:
    try:
        p.fly()
    except CantFlyError:
        p.speak("Sorry")
```

```
for p in row:
    try:
        p.fly()
    except CantFlyError:
        p.speak("Sorry")
```

try/except

You will get more practice with try/except in today's homework.

Python modules (aka packages)

By default, only a handful of the most commonly used functions and object classes are loaded when you start Python.

This saves memory so that your Python scripts can run as quickly as possible.

The object classes and functions that are loaded automatically with Python are called **built-in**.

Python modules (aka packages)

Modules are groups of related functions and object classes

They are not automatically loaded when you start Python

You must **import** the module every time you start Python

If you do not have the module on your computer, you must first **install** it (only once)

Python modules (aka packages)

- Anaconda comes with many of the most common modules pre-installed
- Many others can be installed from the command line
- Some must be downloaded from the creator
- There are modules for every field and use

Python modules (aka packages)

Today's homework will show you how to import packages and work with a couple functions from packages that are included in Anaconda.

TUESDAY HOMEWORK

Complete by 10 am Central tomorrow

Answer keys are in today's folder with the homework and quiz

- tuesdayHW1.ipynb
- tuesdayHW2.ipynb
- tuesdayQuiz.ipynb

The homework contains new skills, it isn't just repetition.