AggieSTAAR Python Bootcamp

Tutorial 1: Python basics and troubleshooting

TEXAS A&M UNIVERSITY

Before you start:

You should have Python installed on your computer, and should also have some program that can run a view notebooks ready to go (Jupyter, Anaconda, VSCode). These slides assume that you are using Jupyter.

If not, go back to and follow Tutorial 0.

To-do:

Open up Jupyter lab through Anaconda, or by typing “jupyter lab” into a terminal, and open up tutorial1_basics.ipynb.

To complete both exercises in the tutorial, you will need to have numpy and astropy installed, as well as hetmgs_tab1_clean.txt downloaded in the same folder as tutorial1_basics.ipynb.

Data types

Data types - there are many, but here are the ones to focus on:

- int (integers),
- float (floating point decimals),
- str (strings).

You can convert in between data types using int(), float(), and str(), and you can check an object's data type using type().

Operations

Operations - again, there are many, but here are the ones to focus on:

- + (addition),
- - (subtraction),
- * (multiplication),
- / (division),
- ** (exponentials).

Be wary of the object's type when trying to perform operations! You cannot multiply strings, for example.

Lists

Lists - the easiest way to store data.

Indexing lists takes a second to get used to - Python counts starting at 0. Also, -1 corresponds to the LAST element.

```
list_example = ['a', 'b', 'c', 'd', 'e']
```

index 0 index 1 index 2 index -1

Lists

Lists - they can be modified by several means:

- indexing (change element at index),
- append (append element to end of list),
- insert (insert element at index),
- + (add one list to the end of another, only works with two lists).

All are suited for different situations - keep experimenting to get a good handle for when to use what.

Lists

Lists - have other operations:

- len(): checks the length of a list.
- Slicing - use the ‘:’ operator while indexing to cut lists between certain indices, see tutorial for more details.

Arrays

Arrays - functionally similar to lists, with more math capabilities.

Import in numpy to begin!

- indexing, appending, inserting: all still work, see tutorial for syntax differences.
- Maths operations: addition, subtraction, multiplication, and division <u>ALL work with arrays</u>!
- numpy has *many* built in functions.

Troubleshooting

Troubleshooting is an important skill to learn - things WILL go wrong!

A lot of errors are self explanatory: IndexError, ModuleNotFoundError, TypeError, ValueError - go through your code, and make sure that it has been written correctly.

print() statements are your best friends! Make sure you have consistency checks in your code.

Troubleshooting – Example A

```
A = ['32', 3, 56.4]

B = [2, '22', 8, 16]

print(A[0] / B[-1])

> TypeError:
unsupported operand
type(s) for /: 'str'
and 'int'
```

What exactly went wrong here?

This might be fairly straightforward to spot, since I've highlighted the problem and the lists are not that long.

But what if your lists are 1000 elements long?

print() statements are your best friends!

Troubleshooting – Example A

```
A = ['32', 3, 56.4]
print(A[0])
B = [2, '22', 8, 16]
print(B[-1])
print(A[0] / B[-1])
> '32'
> 16
> TypeError: unsupported operand type(s) for /: 'str' and 'int'
```

By inserting a few print statements, we can very clearly see that the TypeError arises from A[0] being a string and B[-1] being an integer – you cannot divide a string by an integer.

Troubleshooting

What if the error is NOT obvious? These are sometimes called “semantic errors”

- You’re code executes successfully, but did not perform quite as you expected

This is another reason to have print() statements for sanity checks, they can help you track down the bug

Troubleshooting - Example B

```
X = [2, 4, 6, 7, 23, 45, 25]

# multiply elements and print result

print(X[1]*X[6])

> 100
```

Let's say I wanted to multiply the first and last elements of list X

- Expected answer: 2*25 = 50

But the output was 100, what happened?

Troubleshooting - Example B

```
X = [2, 4, 6, 7, 23, 45, 25]
print(X[1])
> 4 # Should be 2!!
print(X[0])
> 2 # there we go
print(X[6])
> 25 # being extra sure
print(X[0]*X[6])
> 50 # all fixed!
```

Right, python starts counting from 0!

- The first element then is X[0]

I luckily had the correct index for the last element, but I could avoid miscounting by using X[-1] instead.

- This would be useful if the list were super long

Troubleshooting

When in doubt, look it up!

- If you have a bug in your code that just won't go away, look up the error or describe the problem (someone has probably dealt with and fixed it)
  - Keep in mind the operating system you are using (Mac or Windows) when looking up solutions
- Check out package documentation
  - astropy.io.ascii docs
  - It's ok if the documentation is confusing at first, eventually pieces will start making sense :)