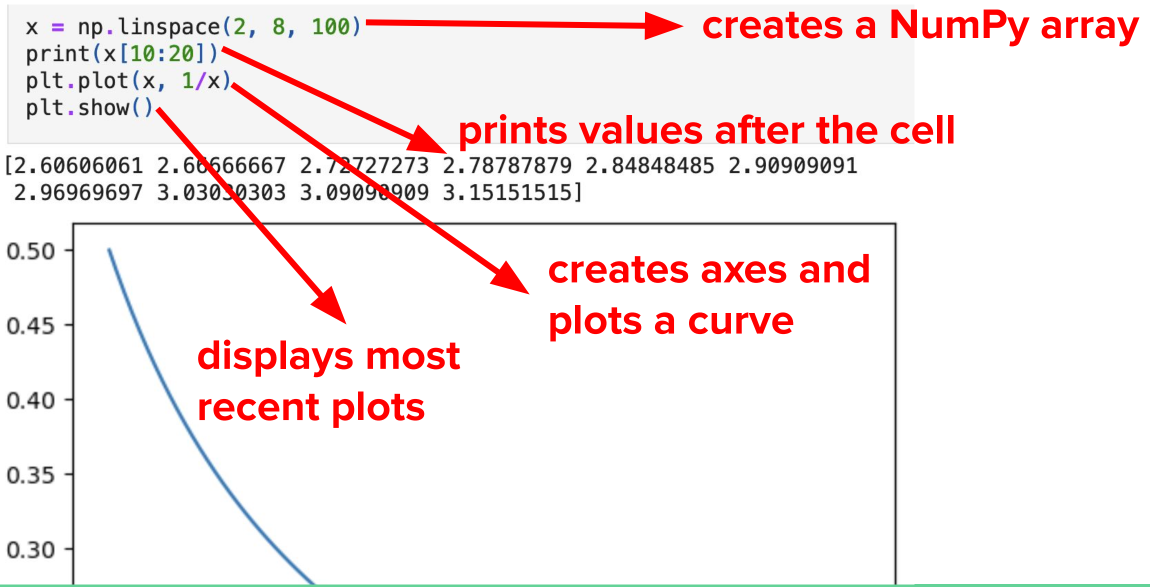
Defining Functions
Day 12 – PH 365

6 Nov 2024

Calling Functions

Connecting a **word** to **parentheses** is how we call functions in Python

```
function_name(input1, input2)
```



```
x = np.linspace(2, 8, 100)
print(x[10:20])
plt.plot(x, 1/x)
plt.show()
```

```
[2.60606061 2.66666667 2.72727273 2.78787879 2.84848485 2.90909091
 2.96969697 3.03030303 3.09090909 3.15151515]
```

creates a NumPy array
prints values after the cell
creates axes and plots a curve
displays most recent plots
0.50
0.45
0.40
0.35
0.30

Defining Functions

Defining a function is like importing a library – we don't have to **call** the function right away, but we enable the computer to have access to it

```
def function_name(input1, input2):
    indented code
    return some_value
```

Function definition

```
new_value = function_name(5, 2.3)
```

Function call

Returning vs Printing values

We use **return** to output a value, **print** to display a value

```
def function_name(input1, input2):
    indented code
    return some_value


new_value = function_name(5, 2.3)
```

Value of `some_value` gets stored in variable **new_value**

Returning vs Printing values

We use **return** to output a value, **print** to display a value

```
def function_name(input1, input2):
    indented code
    print(some_value)


new_value = function_name(5, 2.3)
```

Value of **`some_value`** gets printed, but **nothing** gets stored in variable **new_value**