1 Python: Lecture 1 on 16 May 2020

- 5 Minutes - Welcome to Programming! What is Python good for? Why is it so powerful?
- 10 minutes - Introduction To Repl.It
- Remaining - Go through code concepts, examples, etc.

2 Code

2.1 Getting Started

Listing 1: Using print statements

```python
print("hello, world")
print("hello", ",", "world")
print("hello", "world", sep = ", ")
print("hello", "+"world")
```

2.2 Operations

First up, we learn how to use basic math operations using integers.

Listing 2: Basic math operations

```python
print(3 + 3)  # should be 6
print(3 - 3)  # should be 0
print(3 * 3)  # should be 9
print(3 ** 3) # should be 27
print(3/3)    # should be 1
```

However, we can also use math operations with decimal values.

Listing 3: Decimal-based math operations

```python
print(3.2 + 3.3)
print(10.9 - 9.2)
print(3.6 * 2.5)
```

We can also force integer division.

Listing 4: Forcing integer division

```python
print(7/3)   # prints out 2.3333
print(7//3)  # prints out 2, why?
```

Finally, we can also use the modulus to find the remainder of a number.

Listing 5: Forcing integer division

```python
print(10 % 3) # should be 1, but why?
print(2 % 2)  # if the number is even, mod will result 0, if not, will result 1
```

2.2.1 Task - Distance Formula (Pythag)

Assuming the points (5, 3) and (4, 5) find the distance between the points using the pythag theorem.

Listing 6: Solution

```python
print(( (5-3)**2 + (4-5)**2)**1/2) # note the (** 1/2) is square root.
```

2.3 Variables

Variables are extremely powerful in Python. Being able to know how to use variables is extremely important. The following shows you how to define them.

Listing 7: Basic Variables

```python
first_name = "Bill"
last_name = "Gates"
age = 64
number_pi = 3.141592653589793

print(first_name + last_name)
print(last_name, first_name)
print(last_name, first_name, sep = ", ")
```
On top of that, you can define multiple variables in the same line and use them for the same math operations like above such as addition. However, remember that adding strings is different from adding numbers, as seen below.

### Listing 8: More Variables

```python
n1, n2 = (3, 4)
print(n1 + n2)  # prints 7
n3 = n1 + n2
print(n3)
```

n1, n2 = "3", "4"
print(n1+n2)  # prints 34

Finally, you can also make multiline strings by using 3 quotes in a row for the beginning and end of a string.

### Listing 9: Long Strings

```python
long_string = "The quick brown fox jumped over the lazy old dog."
print(long_string)  # will print the entire string out
```

### 2.3.1 Task: Distance Formula w/ Variables

Implement the same distance formula from before except use variables to define the points and define your solution afterwards.

### Listing 10: Problem

```python
x1, y1 = 5, 3
x2, y2 = 4, 5
solution = ((x1-x2)**2 + (y1-y2)**2)**1/2
print(solution)
```

### 2.3.2 Task: Say Hi to Yourself!

Make two variables to make strings that take in your name, you can customize this to be any name that you would like. Once you do that, make it say “Hi” by using a print statement.

### Listing 11: Solution

```python
first_name = "Jeff"  # enter your name
last_name = "Bezos"  # enter your last name
print("Hey there", first_name, last_name)  # what will this print?
```

### 2.4 Inputs

One of the most fundamental parts of any language is interacting with the user. In this section, we will explore taking input from a user and using it within our code.

### Listing 12: Taking Input from user

```python
first_name = input("Please enter your first name: ")
last_name = input("Please enter your last name: ")
age = input("Please input your age: ")
print("Hey there", first_name, last_name)
print("You are", age, "years old")
```

### 2.4.1 Task: Simple Email Creator

Use an input statement to collect the first and last names of the person running your program, with this, use string concatenation (adding strings) to print out a gmail email formatted as: BillGates@gmail.com if first name was Bill and last name was Gates.

### Listing 13: Solution

```python
first_name = input("Enter your first name")
last_name = input("Enter your last name")
print(first_name+last_name"@gmail.com")
```
2.5 Complex Inputs

Before we get into anything, let’s run an experiment based on taking input from the user.

Listing 14: Experimental Inputs

```python
n1 = input("Please enter a number: ")
n2 = input("Please enter another number: ")
print(n1 - n2) # why do you think this fails?
```

This fails with an error, can you think of a reason why? It’s because when we input anything using the `input` function, it will be read as a string and you cannot subtract strings. We can prove this by using the following code:

Listing 15: Built-in Type Function

```python
#To find out: try the built in command called type().
#This command allows us to find the type of both n1 and n2.
print(type(n1))
print(type(n2))

#Notice that both of these are strings.
#This means that Python is actually interpreting the computation as "3" - "4"
#Or in other words like "Jeff" - "Bezos"
```

In order to solve this, we can use type casting which is where we force a string that looks like a number to act like a number. This is done with the following code:

Listing 16: Parsing Strings as Integers

```python
n1 = int(input("Please enter a number: ")) # int() forces it to be interpreted as an integer
n2 = int(input("Please enter another number: "))
print(n1 - n2)
```

2.5.1 Task: Take an Age and Add 4

Ask the user for their age and add 4 to it, print this back to them

Listing 17: Solution

```python
age = input("What is your age?")
fixed_age = int(age)
print("In 4 years you will be " + str(fixed_age+4))
```