

Lesson 1: Basic Syntax

July 31, 2020

1 Lesson 1: Basic Syntax

In this lesson we will learn about...

- Your First Program: "Hello, World!"
- Identifier Rules and Reserved Words
- Lines and Indentation
- Quotation
- Comments
- Input and Output

2 A Word of Warning about Python Versions

- Two main version of python are still in use: Python **2** and Python **3**.
- We are using Python **3**. Make sure you are using the correct tools!

3 "Hello, World!"

The "Hello, World!" program is the most basic you can run, and is traditionally your first program.

```
[1]: print("Hello, World!")
```

Hello, World!

This program simply writes `Hello, World!` to the screen. We'll go more in-depth about the parts of this later. Use this program to make sure your environment is set up correctly and that you can run Python code.

In IDLE:

0. You may see the python shell pop up. You can run code directly here, which is great for a few lines, but we want to be able to run a whole file of code at once. As such, even though Hello World is one line, we'll use a file.
1. Go to **File > New** to create a new Python file.
2. Type the code into this file, then use **File > Save As...** to save it with the `.py` file extension. Avoid spaces in file names if you can.

3. To run the code, go to **Run > Run Module**. The Python Shell will open and run the code in your file.

4 Indentation

- Blocks of code need to be indented
- The number of tabs/spaces is up to you, but must be consistent
- Python's Style Guide suggests using 4 spaces per indent
- No braces are used, unlike many other programming languages

At this moment, you don't need to know what this code does. Rather, notice how the indentation creates blocks of code.

```
[2]: # Correct
if True:
    print("True")
else:
    print("False")
```

True

```
[3]: # Incorrect
if True:
print("True")
else:
print("False")
```

```
File "<ipython-input-3-71288eeef709>", line 3
print("True")
~
```

IndentationError: expected an indented block

5 Quotation

- Use single ('), double ("), or triple (''' or """) quotes to denote string literals
- Single and Double quotes span one line
- In python, these have no syntactic difference
- Triple quotes can span multiple lines
- Use a backslash (\) to escape a quotation mark in a string
- You do not have to escape " inside ', and vice versa

```
[4]: string1 = 'string'
string2 = "string"
```

```

sentence = "This is a sentence \"with double quotes\" "
sentence2 = 'This is also a sentence "with double quotes in it"'
multiline = """This is a quote
that takes multiple lines. Notice
how the newlines are part of the string too!"""

print(string1)
print(string2)
print(sentence)
print(sentence2)
print(multiline)

```

```

string
string
This is a sentence "with double quotes"
This is also a sentence "with double quotes in it"
This is a quote
that takes multiple lines. Notice
how the newlines are part of the string too!

```

- Did you notice that `multiline` kept the line breaks when printed?
- The new line is just a character (`\n`)!
- When you use multiline strings, it gets inserted wherever you insert a new line.
- You can also manually insert this character into any string by typing `\n`

6 Comments

Comments are pieces of text in a python file that are not interpreted as code.

Use the `#` to denote a comment anywhere in a line. The rest of the line after `#` is considered a comment. Blank lines are also ignored by python.

```

[5]: # This is a comment
a = 2 # This is also a comment

# blank lines also are ignored!

```

7 Input and Output

- Use `input()` to get input from the keyboard.
- Use `print()` to send output to the screen.

```

[6]: text = input("Enter some text: ")
print(text)

```

Enter some text: This is from the keyboard.

This is from the keyboard.

- The argument to `input` (the string between the parenthesis) is the prompt to be shown to the user.
- Always end this with a space.