Basic Syntax

Md. Nahiyan Uddin

Lecture in CSE, UAP

Python Installation

- Before executing the python file, we need to first check if we have python already installed in our device.

- To check, write the following command:

```
$ python --version
```

Hello world!

- Only one line of code is needed in python

```
print "Hello, Python!"
```

- One command to execute the file (suppose the code is in “test.py” file)

```
$ python test.py
```

Writing python codes in google colab

- It is a online platform where we can run our python codes with more ease

- **Colab** notebooks also allow us to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more.

- Go to https://colab.research.google.com/notebooks/intro.ipynb and create your own colab notebook

Python Identifiers

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language. Thus, **Manpower** and **manpower** are two different identifiers in Python.

Python Identifiers

Here are naming conventions for Python identifiers −

1. Class names start with an uppercase letter. All other identifiers start with a lowercase letter.
2. Starting an identifier with a single leading underscore indicates that the identifier is private.
3. Starting an identifier with two leading underscores indicates a strongly private identifier.
4. If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

Reserved Words – words we can not use as identifiers

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

Lines and Indentation

- Python provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is rigidly enforced.
- The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. For example −

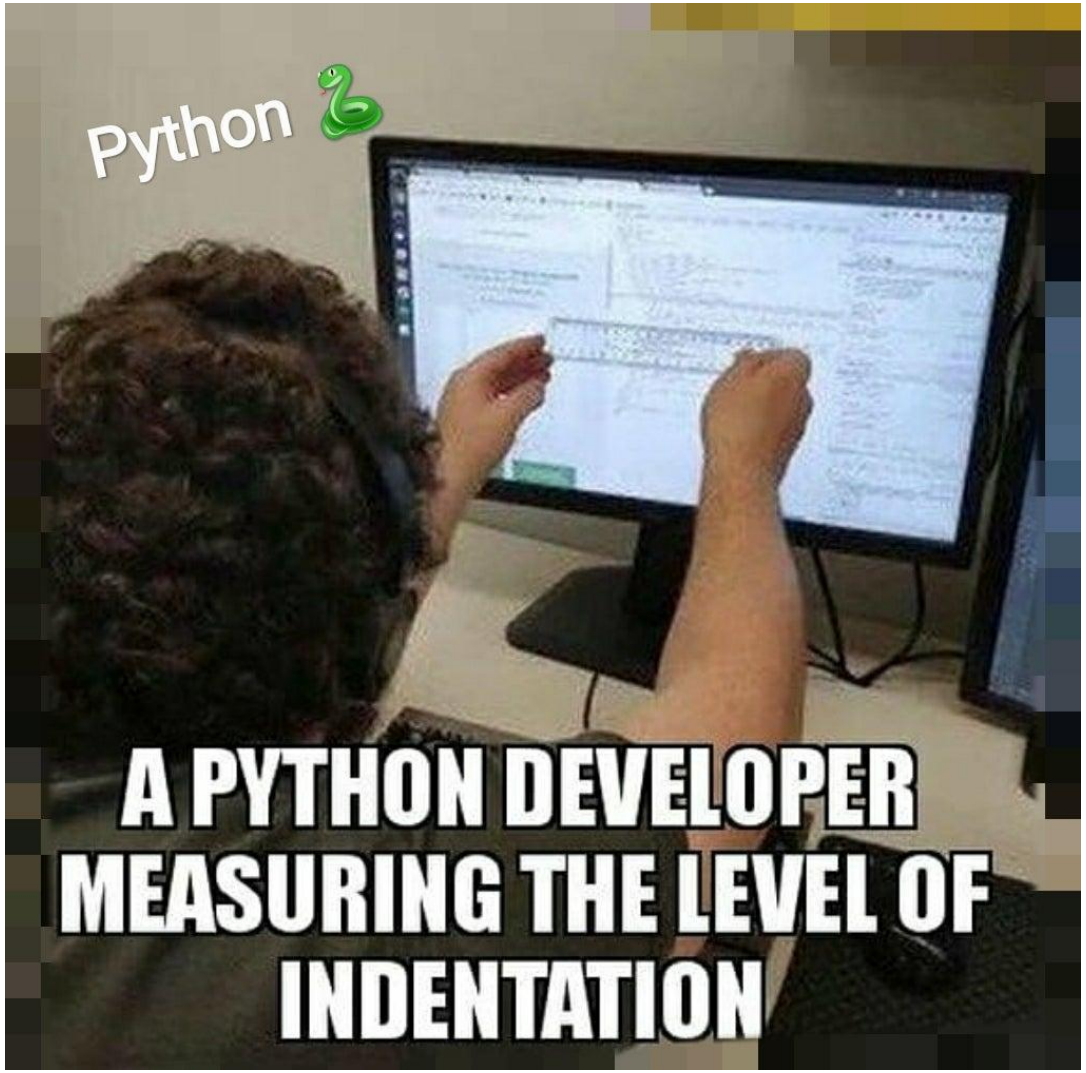
```
if True:
   print "True"
else:
   print "False"
```

Lines and Indentation

- However, the following block generates an error −

```
if True:
print "Answer"
print "True"
else:
print "Answer"
print "False"
```

Lines and Indentation



Multi-Line Statements

- Statements in Python typically end with a new line. Python does, however, allow the use of the line continuation character (\) to denote that the line should continue. For example −

```
total = item_one + \
        item_two + \
        item_three
```

- Statements contained within the [], {}, or () brackets do not need to use the line continuation character. For example −

```
days = ['Monday', 'Tuesday', 'Wednesday',
        'Thursday', 'Friday']
```

Quotation in Python

- Python accepts single ('), double (") and triple (''' or """) quotes to denote string literals, as long as the same type of quote starts and ends the string.

```
word = 'word'
sentence = "This is a sentence."
paragraph = """This is a paragraph. It is
made up of multiple lines and sentences."""
```

Comments in Python

- A hash sign (#) that is not inside a string literal begins a comment. All characters after the # and up to the end of the physical line are part of the comment and the Python interpreter ignores them.

```
#!/usr/bin/python

# First comment
print "Hello, Python!" # second comment
```

```
name = "Madisetti" # This is again comment
```

Comments in Python

- You can comment multiple lines as follows – (use “ctrl+/”)

```
# This is a comment.
# This is a comment, too.
# This is a comment, too.
# I said that already.
```

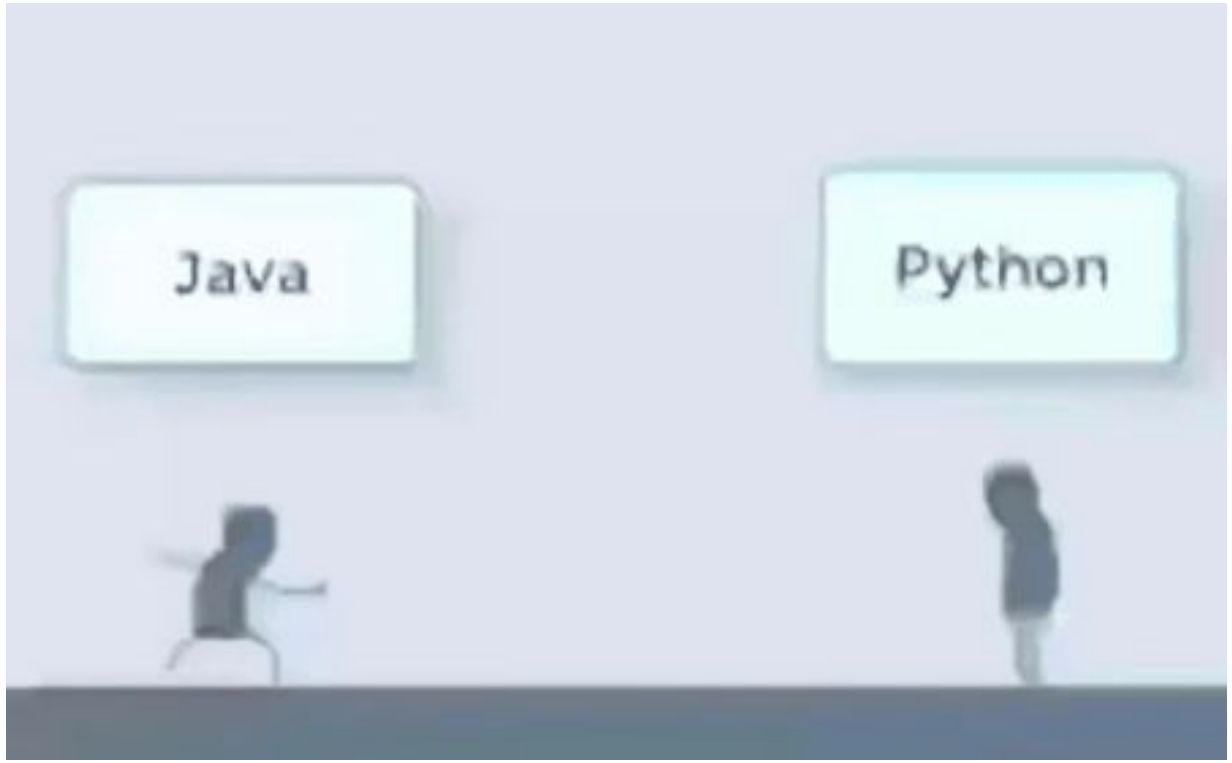
- Or you can use triple quote for a multiline comment

```
'''
This is a multiline
comment.
'''
```

Multiple Statements on a Single Line

```
import sys; x = 'foo'; sys.stdout.write(x + '\n')
```

In summary



Thank you