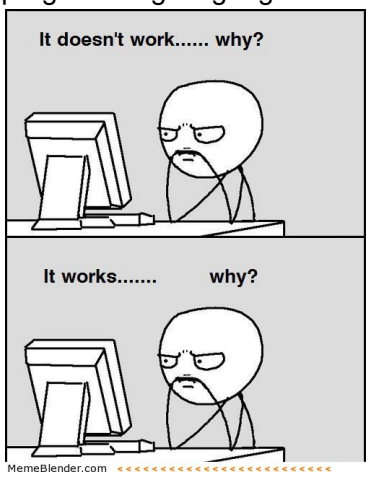
Python Syntax

Objectives:

- Zen of Python Programming
- Identifiers, Keywords and Variables
- Statement and Comment

1. Zen of Python Programming

The Zen of Python is a collection of 19 "guiding principles" for writing computer programs that influence the design of the Python programming language.



Get Zen on REPL

Exercise:

- Start IDLE
- Type following command

```
import this
```

It prints out the 19 principles in REPL:

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Flat is better than nested.
- Sparse is better than dense.
- Readability counts.
- Special cases aren't special enough to break the rules.

- Although practicality beats purity.
- Errors should never pass silently.
- Unless explicitly silenced.
- In the face of ambiguity, refuse the temptation to guess.
- There should be one-- and preferably only one --obvious way to do it.
- Although that way may not be obvious at first unless you're Dutch.
- Now is better than never.
- Although never is often better than *right* now.
- If the implementation is hard to explain, it's a bad idea.
- If the implementation is easy to explain, it may be a good idea.
- Namespaces are one honking great idea -- let's do more of those!

Read up brief explaination of Zen principles: https://medium.com/@Pythonidaer/a-brief-analysis-of-the-zen-of-python-2bfd3b76edbf (https://medium.com/@Pythonidaer/a-brief-analysis-of-the-zen-of-python-2bfd3b76edbf)

2. Identifiers, Keywords and Variables

1.1 Python Identifiers

Python Identifiers are user-defined names to represent a variable, function, class, module or any other object.

Guidelines for Creating Identifiers

- Identifier name can contain following charaters
  - a sequence of letters either in lowercase (a to z) or uppercase (A to Z)
  - digits (0 to 9)
  - underscore (_)
- Identifier name cannot begin with digits
- Special characters are not allowed

Exercise: Type following commands in REPL:

```
message = 'hello'
_message = 'again'
1message = 'failed'
message@ = 'not allowed'
```

Best Practices

- Class names should start with a capital letter. All other identifiers should begin with a lowercase letter.
- Declare private identifiers by using the single-underscore _ as their first letter.
- Don’t use double-underscores __ as the leading and trailing character in an identifier.
  - Python built-in types already use this notation.

- Use underscore _ to join multiple words to form a sensible name.

2.2 Python Keywords

Keywords are special words which are reserved and have a specific meaning. Python has a set of keywords that cannot be used as variables in programs.

Exercise:

```
import keyword
keyword.kwlist
```

The Keywords should NOT be used as identifiers because they are reserved.

```
False      True       None     and      as
assert     async      await    break    class
continue   def        del      elif     else
except     finally    for      from     global
if         import     in       is       lambda
nonlocal   not        or       pass     raise
return     try        while    with     yield
```

Exercise:

```
for = 1
True = 1
```

1.3 Python Variables

A **variable** represents an entity which holds a value.

- Variables in Python don’t require declaration
- Variables must be initialized before use.

Exercise:

```
x = 111
id(x)
x = 222
id(x)
help(id)
```

Question:

What happens under the hood when above commands run?

- A variable `x` is created
- A memory location is used to hold value `111`
- Another memory location is used to hold value `222`
- Variable `x` points to new memory location

Question:

What does functin `id()` do?

- Return the identity of an object.
- CPython uses the object's memory address.

Object Caching

For optimization, Python builds a cache and may reuses some of the immutable objects, such as small integers and strings

Exercise:

```
m = 1
id(m)
n = 1
id(n)
```

Question:

What do you observed in `id` values of `m` and `n` ?

They have same ID value, which shows that `m` and `n` point to same memory location.

3. Python Statement and Comment

3.1 Statement

A **statement** is a logical instruction which Python interpreter can read and execute.

- A statement can be an expression or an assignment statement.

Expression

- Expression is a logical sequence of numbers, strings, objects, and operators.

Exercise:

```
1 + 2 + 3
pow(2, 3)
eval("1+2+3")
```

Assignment Statement

- An assignment statement has the syntax of `variable = expression` .
  - Value is assigned from right-hand-side to left-hand-side

Exercise:

```
exp = "2 + 3"
result = eval(exp)
msg = exp + ' equals to ' + str(result)
print(msg)
```

3.2 Line Continuation

Explicit Line Continuation

- A statement can be extended over to multiple lines using the line continuation character `\`

Exercise:

```
exp = "2 ** 3"
result = eval(exp)
msg = exp + \
    ' equals to ' + \
    str(result)
print(msg)
```

Implicit Line Continuation

- If a statement contains either of parentheses `( )` , brackets `[ ]` and braces `{ }` , Python interpreter will automatically detect the line continuation.

Exercise:

```
val = (1 + 2
    * 3 - 4
    / 5 + 6
    )
print(val)
```

```
arr = [ 1,
    2,
    3]
print(arr)
```

3.3 Indentation

Many high-level programming languages, e.g. C and Java, use braces `{ }` to mark a block of code.

Python does it via indentation.

- Statements in a Python code block have same indentation.
- For example, body of a function or a loop

Exercise:

What's the error message when following code runs?

```
msg = 'hello'
  name = 'world'
```

Exercise:

What does the `get_area()` function do?

```
def get_area(radius):
    area = 3.14 * (radius ** 2)
    return area

get_area(2)
```

Indentation Size

Python requires all statements in a code block have same indentation. An indentation can contains any number of space.

- Python style guidelines (PEP 8) states that you should use 4 spaces for an indentation.
- Google recommends indentation to be 2 spaces.

3.4 Comment

Comments are non-executable part of the code.

- They are commonly used in codes for documentation.
- Writing comments is a good programming practice.

As C or Java uses `\\` and `\* *\` for comments, Python use `#` character.

- Every line of comment in Python must begin with a `#` character.
- Comment must be at the same indent level as the code beneath it.
- In-line comments are comment behind a statement.

Exercise:

Try out following commands in REPL and correct its error.

```
# this is a comment
  x = 3.14 # pi value
print(x)
```

4. Recap

- Why is indentation so important in Python?
- How many spaces is an indentation in Python?
- How to expand a Python statement over multiple lines?
- How to comment off a Python code block?