Python Basics Cheat Sheet

This cheat sheet provides you with all the Python Basics in one place.

Variables

Defining variables

```
miles = 10                # type integer
first_name = 'John'       # type string (the value of the variable is between single quotes)
last_name = "Wick"        # type string (use single or double quotes)
a, b, c = 1.5, 3, 'x'     # defining 3 variables on the same line (float, an integer and a string)

# PEP 8 recommends using snake _case for variable names
max_permitted_value = 500    # snake-case notation
maxPermittedValue = 500      # camel-case notation

# Invalid or not recommended names
4you = 10              # not permitted, name starts with a digit
valu!es = 100          # not permitted, name contains special characters
str = 'Python'         # not recommended, name str is a Python language keyword
_location = 'Europe' # not recommended name.
# Avoid names that start with underscores (they have special meaning)
```

Comments in Python

Comments in Python start with the hash character # and extend to the end of the physical line.
If you want to comment out more lines, insert a hash character at the beginning of each line.

```
# This line is a comment.
```

```
The following line is commented out and will be ignored by the Python Interpreter
# x = 1

a = 7    # defines a variable that stores an integer
my_str = 'A hash character # within a string literal is just a hash character'
```

Data Types

```
age = 31      # type int
miles = 3.4  # type float
finished = True                  # type bool
name = 'Andrei'                  # type str (string)
years = [2018, 2019, 2020]    # type list
week_days = ('Monday', 'Tuesday', 'Wednesday')  # type tuple
vowels = {'a', 'e', 'o', 'u'}                # type set
fs = frozenset((1, 2, 'abc', 'xyz'))   # type frozenset

# type dictionary
countries = {'de':'Germany', 'au':'Australia', 'us':'United States of America', 'gr':'Greece'}
```

Python Operators

Arithmetic Operators

```
a = 9
b = 4
a + b         # addition operator => 13
a - b         # subtraction operator => 5
a * b         # multiplication operator => 36
a / b         # division operator => 2.25
a // b        # floor division operator => 2
5.0 // 3.0    # => 1.0 -> works on floats too
a ** b        # exponentiation operator (a to the power of b) => 6561
a % b         # modulus operator => 1
```

Assignment Operators

```
a = 5
a += 2    # shorthand for a = a + 2 => a = 7
a -= 3    # shorthand for a = a - 3 => a = 4
a /= 2    # shorthand for a = a / 2 => a = 2
a *= 3    # shorthand for a = a * 3 => a = 6
a **=2    # shorthand for a = a ** 2 => a = 36
```

Arithmetic Built-in Function

```
divmod(9, 4)   # returns the quotient and the remainder using integer division => (2, 1)
sum([1,2,4])   # returns the sum of an iterable => 7
min(1,-2,3)    # returns the minimum value => -2
max(1,2,4)     # returns the maximum value => 4
a = 10/3       # a = 3.3333333333
round(a, 4)    # returns a number rounded with 4 decimals => 3.3333
pow(2, 4)      # 2 ** 4 = 16
```

Comparison and Identity Operators

```
# Assignment operator is =
a = 2
b = 3

# Equality operator is ==
# It compares the values stored in variables
a == b    # => False
b == b    # => True

# Inequality operator is !=
a != b    # => True

# Other comparisons
a > b     # => False
5 >= 5    # => True
b <= a    # => False

'Python' == 'python'    # => False, case matters
"Python" == 'Python'    # => True, double and single quotes are equivalent

id(a)    # => returns the address where the value referenced by a is stored. Ex: 140464475242000
```

```
# is operator checks if two variables refer to the same object (saved at the same memory address)
a is b  # => False = compares the address of a to the address of b
# equivalent to:
id(a) == id(b)
```

Boolean Variables

```
# True is 1 and False is 0
True == 1     # => True
bool(True)     # => 1

False == 0      # => True
bool(False)    # => 0

1 is True     # => False
0 is False    # => False

True > False                  # => True
a = (True + True ) * 10    # => 20

id(True)    # => 10714848 (you'll get another value)
id(4 > 2)   # => 10714848  - the address of True and False is constant during program execution

# The next 2 expressions are equivalent
(4 > 2) == True     # => True
(4 > 2) is True     # => True
```