Cheatography

# Python Toolbox Cheat Sheet

by TeunR via cheatography.com/213749/cs/46548/

## Data Types

| | | |
|---|---|---|
| Text Type | str; str() | 'I'm a string' |
| Numeric Types | int; int() | 10 |
| | float; float() | 10.3 |
| Boolean Type | bool | True/False |
| Sequence Types | list; list() | [1, 2, 'a', 'b'] |
| | tuple; tuple() | (1, 2, 3) |
| | range | range(4) |
| Set Type | set; set() | {1, 2, 3} |
| Mapping Type | dict; dict() | {1:'a', 2:'b', 3:'c'} |

## Booleans

### Booleans as Numbers

True==1 False==0

### Comparison Operators

| | | | |
|---|---|---|---|
| a==b | is *a* equal to *b*? | a!=b | is *a* different than *b*? |
| a<b | is *a* less than *b*? | a<=b | is *a* less than or equal to *b*? |
| a>b | is *a* greater than *b*? | a>=b | is *a* greater than or equal to *b*? |

### Membership and Identity Operators

| | | | |
|---|---|---|---|
| a in b | is *a* in *b*? | a is b | are *a* and *b* the same object? |
| a not in b | is *a* not in *b*? | a is not b | are *a* and *b* different objects? |

### Boolean Operators

| | |
|---|---|
| not | returns *False* if operand is True, *True* otherwise |
| and | returns *True* if Both operands are True, *False* otherwise |
| or | returns *False* if both operands are False, *True* otherwise |
| ^ (xor) | returns *False* if both or neither operands are False, *True* otherwise |

## Operator Precedence

| | | |
|---|---|---|
| 1. | () | Parentheses are evaluated first |
| 2. | ** | Exponent |
| 3. | +, - | unary + and - signs (e.g., -x) |
| 4. | *, /, //, % | multiplication, division, floor division, and modulo |
| 5. | +, - | Addition, subtraction |
| 6. | ^ | Bitwise XOR |
| 7. | in, not in, is, is not, <, <=, >, >=, !=, == | Comparison, identity, and membership operators |
| 8. | not | logical NOT |
| 9. | and | Logical AND |
| 10. | or | logical OR |

## Print Function

```
print('a', 'b', sep='*')  a*b
```

## Decision Structure

```
if n == 0:
  print( "n is zero")
elif n > 0:
  print( "n is strictly positi ve")
else: # n < 0
  print( "n is strictly negati ve")
```

## Repetition Structures

```
n = 0
while n < 4
  print(n)
  n += 1
print( "n =", n)
# output is: 0 | 1 | 2 | 3 | n = 4
for i in range(4):
  print(i)
print( "i =", i)
# output is: 0 | 1 | 2 | 3 | i = 3
```




Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Cheatography

# Python Toolbox Cheat Sheet

by TeunR via cheatography.com/213749/cs/46548/

## Exceptions

```
try:
  # run this code
except NameOf Err orT ype1:
  # handle error type 2
except NameOf Err orT ype2:
  # handle error type 2
except:
  # handle any other error
else:
  # run this code if no error
finally:
  # always run this code
```

**Built-in Exceptions**

FileNotFoundError
IndexError
KeyError
ModuleNotFoundError
NameError
SyntaxError
TypeError
ValueError
ZeroDivisionError

## NumPy

```
import numpy as np
```

### Creating arrays

| | |
|---|---|
| np.array([1,2,3]) | Convert python list to NumPy array |
| np.arange(1,5) | Return sequence from start (incl.) to end (excl.) |
| np.arange(1,5,2) | Return stepped sequence from start (incl.) to end (excl.) |
| np.repeat([1,3,-6],3) | Repeat values n times: [1,1,1,3,...] |
| np.tile([1,3,6],3) | Repeat values n times: [1,3,6,1,...] |

### Math functions and methods

All functions take an array as the input




Sponsored by **Readable.com**
Measure your website readability!
https://readable.com