

BASIC BUILT-IN FUNCTIONS	
Function	Explanation
<code>print()</code>	Print text to the console.
<code>input()</code>	Get keyboard input via the console.
<code>import</code>	Announce the use of a specific library in our program.
<code>int()</code>	Convert a string to a numerical value of data type integer.
<code>float()</code>	Convert a string to a numerical value of data type float.
...	..
...	...

VARIABLES	
Command	Explanation
<code>x = 1</code>	Declaration of a variable <b>x</b> with a value of type integer.
<code>PI = 3.14159</code>	Declaration of a constant PI with a value of type float.
<code>name = "Anna"</code>	Declaration of a variable with a value of type string.
<code>is_over_18 = True</code>	Declaration of a variable with a value of type boolean.
<code>ipcon = IPConnection()</code>	Declaration of a variable holding a complex object.


LOGICAL OPERATORS IN CONDITIONS	
Operator	Explanation
<code>=</code>	Assign a value to a variable (not a logical operator!).
<code>==</code>	Compare two values and return <b>True</b> if they are equal.
<code>&gt;</code> or <code>&gt;=</code>	Compares two values and returns true if the left is greater than (or equal to) the right.
<code>&lt;</code> or <code>&lt;=</code>	Compares two values and returns true if the left is less than (or equal to) the right.
<code>!=</code>	Compares two values if they are not equal.
<code>and</code> , <code>or</code>	These operators logically combine two conditions.
<code>not</code>	Negates the expression's result.
<code>x is None</code>	Check if a value if <b>x</b> is <b>None</b> , i.e., not existing.

MATH OPERATORS	
Operator	Explanation
<code>+</code> <code>-</code> <code>/</code> <code>*</code>	The basic arithmetic operators.
<code>%</code> <code>//</code>	Calculates the modulo/integer division of two numbers.
<code>2**3</code>	Calculates 2 to the power of 3.
<code>pow(a, b)</code>	Calculates <b>a</b> to the power of <b>b</b> .

STRINGS	
Expression	Explanation
<code>f"Pi is {PI}"</code>	Format string with placeholders in curly braces.
<code>name[1]</code>	Returns the second symbol from the string <b>name</b> .
<code>len(name)</code>	Returns the number of characters in the string.
<code>"n" in Anna</code>	Checks if the string on the left is contained in the string on the right.
<code>name.strip()</code>	Remove whitespace at the beginning/end of a string.
<code>name.upper()</code>	Make all letters uppercase.
<code>name.lower()</code>	Make all letters lowercase.

This **Python Cheat Sheet** was created at the University of Applied Sciences in Osnabrueck. You are free to share it publicly.

Prof. Dr. Nicolas Meseth



TIME	
Function	Explanation
<code>time.sleep(1.5)</code>	Wait for 1.5 seconds.
<code>time.time()</code>	Get the current date/time as a UNIX timestamp.

RANDOM	
Function	Explanation
<code>random.randint(0, 10)</code>	Generate a pseudo-random whole number between 0 and 10.
<code>random.random()</code>	Generate a pseudo-random real number between 0 and 1.
<code>random.choice(list_a)</code>	Choose a random element from <b>list_a</b> , which must be a list/array.

MATH	
Operator	Explanation
<code>math.sqrt(a)</code>	Calculate the square root from <b>a</b> .
<code>math.pi</code> , <code>math.e</code>	Get the value of the mathematical constants.
<code>round()</code>	Round to the nearest integer.
<code>math.floor()</code> <code>math.ceil()</code>	Round up or down.
<code>math.log()</code>	Natural logarithm
<code>math.log(x, base)</code>	Logarithm to base.

CONDITIONALS	
Command	Explanation
<code>if x == 1:</code>	Check if <b>x</b> is one and execute the code block if true.
<code>elif x == 2:</code>	Add another case to an existing <b>if</b> -statement.
<code>else:</code>	Default code branch if all conditions are false.

CUSTOM FUNCTIONS	
Operator	Explanation
<code>def my_func(x):</code>	Define a function <b>my_func</b> with one parameter <b>x</b> .
<code>return x**2</code>	Return a result from a function to the caller.
<code>def power(exp=2):</code>	Define a function with one parameter that has a default value.

LOOPS	
Operator	Explanation
<code>while a &lt; b:</code>	Repeat code as long as <b>a</b> is less than <b>b</b> .
<code>for i in range(10):</code>	Repeat 10 times (for values <b>i</b> = 0...9).
<code>break</code>	Leave the loop immediately.

LISTS	
Expression	Explanation
<code>vals = [1, 2, 3]</code>	Create a list with three elements.
<code>vals[0]</code>	Get the first element from <b>vals</b> .
<code>vals.append(5)</code>	Append an element (5) to the list <b>vals</b> .
<code>vals[0:2]</code>	Get the first three elements from <b>vals</b> .
<code>len(vals)</code>	Get the number of elements in the list.
<code>vals.pop()</code>	Get the first element and remove it from <b>vals</b> .
<code>for v in vals:</code> <code>print(v)</code>	Iterate through all elements in <b>vals</b> and store the current element on <b>v</b> . Print each element.

DATE & TIME	
Command	Explanation
<code>datetime.now()</code>	Current date and time in ISO-format.
<code>datetime.now().timestamp()</code>	Date and time as UNIX timestamp.
<code>strftime("%H:%M:%S")</code>	Format a string as date/time