Python Beginner Cheat Sheet - Just the Basics

Basic Types of Data

```
   boolean = True / False
   integer = 5
     float = 5.23 (or even 5.0)
    string = "abcd1234"

      list = [val1, val2, val3,…]
dictionary = {key1:val1, key2:val2,…}
     tuple = (val1, val2, val3,…)
```

Math

`+`	addition
`-`	subtraction
`*`	multiplication
`**`	exponent
`/`	division
`//`	integer division
`%`	modulus (remainder)

Math Assignment

`+=`	addition
`-=`	subtraction
`*=`	multiplication
`**=`	exponent
`/=`	division
`//=`	integer division
`%=`	modulus (remainder)

Comparisons

`<`	less than
`<=`	less than or equal to
`>`	greater than
`>=`	greater than or equal to
`==`	equal to
`!=`	not equal to

Boolean Logic

`not`	logical NOT
`and`	logical AND
`or`	logical OR

Make Decisions

```
if <something is True>:
    # execute this
    # indented code
elif <something else is True>:
    # execute this
    # indented code
else:
    # execute this code if the
    # if's and elif's were
    # all False

if <value> in <list>:
    # execute this
    # indented code

if <value> not in <list>:
    # execute this
    # indented code
```

Loops

```
while <something is True>:
    # execute this
    # indented code

for <variable> in <list>:
    # execute this
    # indented code

for <variable> in range(stop):
    # execute this indented code, with
    # <variable> ranging from 0 up to stop

for <variable> in range(start,stop,step):
    # execute this indented code, with
    # <variable> ranging from start up to
    # stop, skipping step amount each time

for <key>, <value> in dict.items():
    # execute this
    # indented code
```

`break`	stops the current loop
`continue`	jumps to the next loop iteration
`pass`	does nothing, but can be used as placeholder code

Functions

```
def function_name(param1, param2, …):
    # execute this
    # indented code
    return <value>
```

File Access

`f = open("file.txt", "r")`	open file for reading
`f = open("file.txt", "w")`	open file for writing
`f = open("file.txt", "a")`	open file for appending
`data = f.read()`	read the file as a string
`data = f.readline()`	read one line as a string
`f.write(data)`	write a string to the file
`f.close()`	close the file

```
with open("file.txt", "r") as f:
    # execute this indented code
    # with f holding the handle to the
    # opened file, then automatically
    # close the file when this indented
    # code is finished
```

Created by Dylan Bennett ~ Last modified: July 1, 2025

Python Beginner Cheat Sheet - Just the Basics

Lists

`l = []`	Create an empty list
`l[i] = x`	Store `x` at position `i`
`l.append(x)`	Add value `x` to the end of the list
`l.extend(k)`	Add list `k` to the end of the list
`l.insert(i, x)`	Insert `x` at position `i`
`l.remove(x))`	Delete the first instance of `x`
`l.pop()`	Delete the first item, returning its value
`l.pop(i)`	Delete the item at `i`, returning its value
`l.clear()`	Delete all items in the list
`l.index(x)`	Return the first position of `x`
`l.count(x)`	Return the number of occurrences of `x`
`l.copy()`	Return a copy of the list
`l.sort()`	Sort the list in ascending order
`l.reverse()`	Reverse the order of the list

Common String Functions

`s.upper()`	Return uppercase version of `s`
`s.lower()`	Return lowercase version of `s`
`s.capitalize()`	Return capitalized version of `s`
`s.title()`	Return title case version of `s`
`s.count(x)`	Count how many times `x` appears in `s`
`s.find(x)`	Return index position of `x` in `s` (or `-1`)
`s.replace(x,y)`	Return version of `s` with `x` replaced by `y`
`s.join(l)`	Return `l` as a single string joined by `s`
`s.isalpha()`	Return whether `s` is all letters
`s.isnumeric()`	Return whether `s` is a number
`s.strip()`	Return version of `s` with whitespace stripped away from the beginning and end
`s.split(x)`	Splits `s` using `x` as a delimiter, or uses spaces if `x` isn't specified, and returns the resulting list of substrings
`s.splitlines()`	Splits `s` at every new line, and returns a list with each line as one item in the list

Common Functions

`print(x)`	Print `x` to the console
`input(s)`	Print `s`, then return user input after ENTER
`int(x)`	Return the integer version of `x`
`str(x)`	Return the string version of `x`
`float(x)`	Return the float version of `x`
`range(i)`	Return numbers from `0` *up to* `i` (but not `i`)
`range(i,j,k)`	Return numbers from `i` *up to* `j`, every `k` value
`len(l)`	Return the number of items in `l`
`sorted(l)`	Return a sorted version of `l`
`min(a,b,…)`	Return the minimum value of `a,b,…`
`min(l)`	Return the smallest value from `l`
`max(a,b,…)`	Return the maximum value of `a,b,…`
`max(l)`	Return the smallest value from `l`
`sum(l)`	Return the sum of all values in `l`
`abs(x)`	Return the absolute value of `x`
`round(x,n)`	Return `x` rounded to `n` digits
`pow(b,e)`	Return `b` raised to the power `e`

List/String Access and Slicing

`l[i]`	Get item/character at position `i`
`l[-1]`	Get the last item/character
`l[beginning:end:step]`	
`l[i:j]`	Get items/characters from `i` *up to* `j` (not `j`)
`l[i:]`	Get items/characters from `i` to the end
`l[:i]`	Get items/characters from position `0` *up to* `i` (not `i`)
`l[::3]`	Get every third item/character from beginning to end
`l[::-2]`	Get every other item/character, but in reverse order

Dictionaries

`dict = {}`	Create an empty dictionary
`dict[k]`	Get the value stored with key `k`
`dict[k] = x`	Store the value `x` using key `k`
`dict.update(d)`	Add the key/values of dictionary `d`
`dict.keys()`	Return a list of keys
`dict.values()`	Return a list of values
`dict.items()`	Return a list of `(key, value)` tuples
`dict.get(k)`	Return the value stored with key `k`
`dict.copy()`	Return a copy of the dictionary
`dict.pop(k)`	Return the value of key `k`, and delete `k`
`dict.clear()`	Delete all items in the dictionary

Modules

```
import module_name
module_name.function_name()

import module_name as other_name
other_name.function_name()

from module_name import *
function_name()

from module_name import function_name
function_name()
```

Created by Dylan Bennett ~ Last modified: July 1, 2025