Python Programming Cheat Sheet

Complete Edition | Calibri + Consolas | Colorized examples

Nikhil Learn Hub | nikhillearnhub.com

Code color legend

`keyword` `string` `number` `comment` `@decorator`

1. BASICS - PRINT, INPUT, COMMENTS

Item	Role
print(...)	Write to stdout
input(prompt)	Read a line (str)
# comment	Line comment
""" docstring """	Multi-line / docs

Syntax

```
print(value, ..., sep=" ", end="\n")
name = input("Prompt: ")
```

Short example

```
print("Hello Python!")
name = input("Name: ")
print(f"Hi, {name}")  # f-string
# This is a comment
```

Key points

- Indentation defines blocks (usually 4 spaces).
- Python is dynamically typed.

2. VARIABLES & DATA TYPES

Type	Example	Notes
int	x = 10	Arbitrary precision
float	pi = 3.14	Decimals
bool	ok = True	True / False
str	s = "hi"	Immutable text
NoneType	x = None	No value
list / tuple / dict / set	See later	Collections

Syntax

```
name = value
type(x)  isinstance(x, int)
```

Short example

```
age = 21
pi = 3.14
flag = True
msg = "learn"
print(type(age), isinstance(pi, float))
```

3. TYPE CASTING & CHECKING

Call	Result
int('5')	5
float(5)	5.0
str(3.2)	'3.2'
bool(0) / bool(1)	False / True
list('ab')	['a', 'b']

Syntax

```
int(x)  float(x)  str(x)  bool(x)
```

Short example

```
n = int(input('n: '))
print(float(n), str(n), bool(n))
```

4. OPERATORS

Kind	Operators	Notes
Arithmetic	+ - * / // % **	// floor div
Comparison	== != < > <= >=	Compare values
Logical	and or not	Short-circuit
Assignment	= += -= *= /=	Update names
Membership	in not in	Containers / str
Identity	is is not	Same object
Bitwise	& | ^ ~ << >>	Integers

Syntax

```
a // b   a ** b   x in seq   a is b
```

Short example

```
print(7 // 2, 2 ** 3)   # 3  8
print(3 in [1, 2, 3])
a = [1]; b = a; print(a is b)
```

5. CONTROL FLOW

Form	Use
if / elif / else	Branching
match / case (3.10+)	Structural pattern match
ternary	a if cond else b

Syntax

```
if cond:
    ...
elif cond2:
    ...
else:
    ...
```

Short example

```
x = 10
if x > 0:
    print("positive")
else:
    print("not positive")
label = "hi" if x > 5 else "lo"
```

6. LOOPS

Loop	Notes
for x in iterable	Iterate items
for i in range(n)	0 .. n-1
while cond	Repeat while true
break / continue	Exit / next
else on loop	Runs if no break

Syntax

```
for x in iterable:
    ...
while cond:
    ...
```

Short example

```
for i in range(1, 4):
    print(i, end=' ')
n = 3
while n:
    print(n); n -= 1
```

7. FUNCTIONS

Feature	Notes
def name(params):	Define function
return	Send value back (or None)
*args / **kwargs	Var positional / keyword
Default args	def f(a, b=0):
lambda	Small anonymous function

Syntax

```
def name(a, b=0, *args, **kwargs):
    return value
```

Short example

```
def add(a, b=0):
    return a + b
print(add(2, 3))
sq = lambda x: x * x
print(sq(4))
```

8. STRINGS

Method / idea	Role
s[i] / s[a:b]	Index / slice
len(s)	Length
upper/lower/strip	Case / trim
split / join	Split / combine
replace / find	Edit / search
f'{expr}'	Format string

Syntax

```
s = "text"
s[start:stop:step]
```

Short example

```
s = "  Python  "
print(s.strip().upper())
print("-".join(["a", "b"]))
print(f"len={len(s.strip())}")
```

Key points

- Strings are immutable.

9. LISTS

Op / method	Role
append / extend	Add items
insert / pop / remove	Insert / delete
sort / sorted	In-place / new list
[i] / [a:b]	Index / slice
list comp	[expr for x in it if ...]

Syntax

```
lst = [1, 2, 3]
lst.append(4)
```

Short example

```
nums = [3, 1, 2]
nums.append(4)
nums.sort()
squares = [x*x for x in nums if x > 1]
print(nums, squares)
```

10. TUPLES

Feature	Notes
Immutable	Cannot change items
Packing	t = (1, 2)
Unpacking	a, b = t
Single item	t = (1,) # comma needed

Syntax

```
t = (a, b)
x, y = t
```

Short example

```
point = (3, 4)
x, y = point
print(x, y, len(point))
```

11. DICTIONARIES

Op / method	Role
d[key] / d.get(k)	Access (get safe)
d[k] = v	Insert / update
keys / values / items	Views
pop / del	Remove
{k:v for ...}	Dict comprehension

Syntax

```
d = {"a": 1}
d.get("a", 0)
```

Short example

```
ages = {"Ann": 20, "Bob": 22}
ages["Cara"] = 19
print(ages.get("Ann"), list(ages.keys()))
```

12. SETS

Op	Meaning
add / discard / remove	Mutate set
| & - ^	Union / intersect / diff / xor
in	Membership (fast avg)
set comp	{x for x in it}

Syntax

```
s = {1, 2, 3}
s.add(4)
```

Short example

```
a = {1, 2, 3}
b = {3, 4}
print(a | b, a & b, a - b)
```

Key points

- Unordered, unique elements. Elements must be hashable.

13. BUILT-IN HELPERS

Function	Use
len / sum / min / max	Aggregate
enumerate / zip	Index pairs / parallel
map / filter	Transform / keep
sorted / reversed	Order views
any / all	Truth over iterable
range(start, stop, step)	Integer sequence

Syntax

```
for i, v in enumerate(seq):
    ...
```

Short example

```
nums = [10, 20, 30]
for i, v in enumerate(nums):
    print(i, v)
print(list(zip([1,2], ['a','b'])))
```

14. OOP - CLASS & OBJECT

Part	Role
class Name:	Define type
__init__	Constructor
self	Instance reference
obj.attr / obj.method()	Use object

Syntax

```
class Name:
    def __init__(self, ...):
        ...
    def method(self):
        ...
```

Short example

```
class Box:
    def __init__(self, w):
        self.w = w
    def area(self):
        return self.w * self.w
b = Box(4)
print(b.area())
```

15. OOP - INHERITANCE & SPECIAL

Idea	Notes
class D(B):	Inherit from B
super()	Call parent
@staticmethod	No self/cls
@classmethod	Receives cls
@property	Attribute-style getter

Syntax

```
class Dog(Animal):
    def speak(self):
        ...
```

Short example

```
class Animal:
    def speak(self):
        return "?"
class Dog(Animal):
    def speak(self):
        return "woof"
print(Dog().speak())
```

16. EXCEPTION HANDLING

Keyword	Role
try	Guard code
except Type	Handle error
else	If no exception
finally	Always runs
raise	Throw error

Syntax

```
try:
    ...
except Exception as e:
    ...
finally:
    ...
```

Short example

```
try:
    n = int('x')
except ValueError as e:
    print("bad", e)
finally:
    print("done")
```

17. FILE HANDLING

Mode	Meaning
'r'	Read text
'w'	Write (truncate)
'a'	Append
'r+'	Read + write
'b'	Binary (e.g. 'rb')

Syntax

```
with open(path, mode) as f:
    ...
```

Short example

```
with open("out.txt", "w") as f:
    f.write("score=95\n")
with open("out.txt") as f:
    print(f.read())
```

Key points

- Prefer with open(...) so files close automatically.

18. MODULES & IMPORTS

Form	Meaning
import math	Import module
from math import sqrt	Import name
import math as m	Alias
if __name__ == '__main__':	Script entry guard

Syntax

```
import module
from module import name
```

Short example

```
import math
from math import sqrt
print(math.pi, sqrt(9))
if __name__ == '__main__':
    print("run as script")
```

19. DECORATORS & COMPREHENSIONS

Pattern	Idea
@decorator	Wrap a function/class
[x for x in it]	List comp
{k:v for ...}	Dict comp
{x for x in it}	Set comp
(x for x in it)	Generator expr

Syntax

```
@decorator
def f(...):
    ...
```

Short example

```
def twice(fn):
    def wrap(x):
        return fn(fn(x))
    return wrap
@twice
def inc(x):
    return x + 1
print(inc(1))  # 3
```

20. REGEX (re) QUICK REF

Call	Role
re.search(pat, s)	First match anywhere
re.match(pat, s)	Match at start
re.findall(pat, s)	All matches list
re.sub(pat, repl, s)	Replace

Syntax

```
import re
re.findall(r'\d+', text)
```

Short example

```
import re
text = 'a12 b34'
print(re.findall(r'\d+', text))  # ['12','34']
```

21. USEFUL STANDARD LIBRARY

Module	Common use
os / pathlib	Paths, files, env
sys	argv, exit, stdin
json	load / dump JSON
datetime	Dates and times
collections	Counter, defaultdict, deque
itertools	Product, chain, groupby
math / random	Math / RNG

Syntax

```
import json
data = json.loads(text)
```

Short example

```
import json
s = '{"n": 3}'
print(json.loads(s)['n'])
```

22. INTERVIEW HOTSPOTS

Topic	Revise
Mutable vs immutable	list vs tuple/str
List vs dict vs set	When to use each
Comprehensions / generators	Memory & style
*args **kwargs / decorators	Flexible APIs
OOP + MRO	Inheritance, super
Exceptions + with	Clean error / resource handling
GIL (CPython)	Threading vs multiprocessing

Syntax

```
[x for x in it if cond]   # list
(x for x in it if cond)   # generator
```

Short example

```
def demo(a, *args, **kwargs):
    return a, args, kwargs
print(demo(1, 2, 3, x=9))
```

Educational summary for Nikhil Learn Hub. Topic map inspired by GeeksforGeeks Python Cheat Sheet; wording is original.