

1 Python CheatSheet

LANGUAGES

- PDF Link: [cheatsheet-python-A4.pdf](#), Category: languages
- Blog URL: <https://cheatsheet.dennyzhang.com/cheatsheet-python-A4>
- Related posts: [Golang CheatSheet](#), [Ruby CheatSheet](#), [#denny-cheatsheets](#)

File me Issues or star this repo.

See more CheatSheets from Denny: [#denny-cheatsheets](#)

1.1 Python Compact Coding

Name	Comment
Return if.. else	<code>return val if i>0 else 0</code>
Multiple assignment	<code>l, r = 2, 3</code>
Assign with check of none	<code>a = b if b else 1</code>
Assignments	<code>l[1]=l[0]=0</code>
Swap values	<code>left, right = right, left</code>
List Comprehensions	<code>[x*x for x in range(1, 1001)]</code>
List Comprehensions	<code>l = [2, 3, 5]; [2*x for x in l if x>2]</code>
Use zip	<code>for a, b in zip(nums, nums[3:])</code>
Build a list	<code>dp = [1] + [0]*3</code>
Change interger to string in binary	<code>bin(num), f'{{num:b}}', "{{0:b}}".format(num)</code>
Sum a subarray	<code>sum(nums[0:k])</code>
Sort list in descending order	<code>sorted(nums, reverse=True)</code>
Dictionary with defaults	<code>m = collections.defaultdict(lambda: 1)</code>
Loop with single statement	<code>while p.left: p = p.left</code>
Print multiple values	<code>print(x, y)</code>
Get both index and item	<code>for i, ch in enumerate(["a", "b", "c"]): print(i, ch)</code>
Mod negative	<code>(-2)%5</code>
Compare values	<code>if 0<=i<n and 0<=j<m and grid[i][j]</code>
if ... return	<code>if k == 0: return False</code>
if... continue	<code>if index == icol: continue</code>
List comprehensive	<code>areas = [dfs(i, j) for i in range(m) for j in range(n) if grid[i][j]]</code>
Python assertion	<code>assert [1,2]==[1,2]</code>

1.2 Python Advanced: Concepts & Internals

Name	Comment
Python Global Interpreter Lock	For Garbage Collection. A mutex controls of the global Python interpreter
Python tuples VS lists	tuple is immutable
Python nonlocal VS global	Github: cheatsheet-python-A4/code/varNonlocalGlobal.py
Python For VS While Loops	The for statement is used to iterate over the elements of a sequence
subprocess.run VS os.system	In Linux, launch processes through shell or os.execvp
single quote VS double quote	Generally double quotes for string; single quotes for regexp, dict keys, or SQL
Common reasons of python memory leak	reference cycles, underly libraries/C extensions, lingering large objects not released
Example: Python cycle reference	Github: cheatsheet-python-A4/code/exampleCycleReference.py
Passing function as an argument in Python	Github: cheatsheet-python-A4/code/funcAsParameter.py
lambda/an anonymous function	
Why no support for multi-line comments	Link: Python Multi-line Comments
Python callable	<code>print(callable(1)), print(callable(lambda: 1))</code>
Python long	
Python Constants vs Literals	
How functools.lru _{cache} works	
Python yield	
Reference	Link: Python Design and History FAQ

1.3 List & Tuples

Name	Comment
Create a fixed size array	<code>[None]*5</code>
Create a fixed size matrix/2D array	<code>[[sys.maxsize for j in range(2)] for i in range(3)]</code>
Flatten 2D array into 1D array	<code>[a for r in matrix for a in r]</code>
Iterate over a list	<code>for v in l:</code>
Iterate over a list with index+val	<code>for i, v in enumerate(l):</code>
zip two lists as one	<code>l = sorted(zip(nums, range(len(nums))))</code>
Convert int array to a string	<code>' '.join([str(v) for v in [1, 2,3,4]])</code>
Extact columns from multi-dimensional array	<code>[row[i] for row in l]</code>
Sort in descending	<code>l=sorted([8, 2, 5], reverse=True)</code>
Sort list by a lambda key	<code>l=sorted([('ebb',12),('abc',14)], key=lambda x: x[1])</code>
Sort list by a function	<code>sorted(logs, key=getKey), LeetCode: Reorder Data in Log Files</code>
In-place sort	<code>l.sort()</code>
Find the index of one item	<code>[1,2,5,3].index(2)</code>
Return all but last	<code>list[:-1]</code>
The second last item	<code>list[-2] or list[~1]</code>
Generate a-z	<code>map(chr, range(ord('a'), ord('z')+1))</code>
Convert from ascii to character	<code>chr(ord('a'))</code>
Reverse a list	<code>["ab", "cd", "ef"][::-1]</code>
map	<code>map(lambda x: str(x), [1, 2, 3])</code>
Copy a range to another range	<code>nums1[k+1] = nums2[j+1]</code>
append an element	<code>array.append(var)</code>
insert elements to head	<code>array.insert(0,var)</code>
delete element by index	<code>del a[1]</code>
list as stack	<code>item = l.pop()</code>
map/reduce	<code>functools.reduce((lambda x, y: "%s %s" % (x, y)), l)</code>
replace ith to jth	<code>list[i:j] = otherlist</code>
combine two list	<code>list1 + list2</code>
get sum	<code>sum(list)</code>
unique list	<code>set(["Blah", "foo", "foo", 1, 1, 2, 3])</code>
Insert to sorted list	<code>bisect.insort(l, 3)</code>
Reverse a list	<code>l[::-1]</code>
Print zip array	<code>print(list(zip(l1, l2)))</code>
Reference	Link: Lists and Tuples in Python

1.4 String

Name	Comment
Reverse string	<code>'hello world'[::-1]</code>
Array to string	<code>' '.join(['a', 'b'])</code>
Integer array to string	<code>' '.join([str(v) for v in [1, 2, 3]])</code>
Split string to array	<code>"hello, python".split(",")</code>
String to array	<code>list('abc')</code>
Format to 2 digits	<code>print "%02d" % (13)</code>
Capitalize string	<code>'hello world'.capitalize()</code>
Upper/lower string	<code>'aBc'.upper(), 'aBc'.lower()</code>
Check if string represent integer	<code>'123'.isdigit()</code>
Check if string alphabetic	<code>'aBc'.isalpha()</code>
Check if string alphanumeric	<code>'a1b'.isalnum()</code>
Count substring	<code>'2-5g-3-J'.count('-')</code>
Remove trailing '0'	<code>'0023'.rstrip('0')</code>
Remove leading '0'	<code>'0023'.lstrip('0')</code>
Trip a string	<code>' Hello '.strip()</code>
Find location of substring	<code>'abc'.find('d')= (returns -1)</code>
Find location of substring	<code>'abc'.index('d')= (raise exception)</code>
Check whether substring	<code>"el" in "hello world"</code>
Replace string	<code>'ab cd'.replace(' ', '')</code>
Padding leading zero	<code>'101'.zfill(10)</code>
Padding whitespace to the left	<code>'a'.ljust(10, '=')</code>
Padding whitespace to the right	<code>'a'.rjust(10, '=')</code>
Format string	<code>"%s,%d,%s" % ("2012", 12, "12")</code>

1.5 Stack & Queue

Name	Comment
Python deque as a stack	<code>s = collections.deque(), s.append(x), s.pop(), s[-1]</code>
Python deque as a queue	<code>s = collections.deque(), s.append(x), s.popleft(), s[0]</code>
Implement a stack in Python	Link: Stack in Python. Leverage: list, collections.deque or queue.LifoQueue

1.6 Python Basic

Name	Comment
Install python3 in Ubuntu	<code>add-apt-repository ppa:deadsnakes/ppa, apt install python3.7</code>
Python constants	
Python nested function	Github: cheatsheet-python-A4/code/nestedFunction.py
Run python snippet from shell	<code>python -c 'import sys; print(sys.getdefaultencoding())'</code>
What's Python Literals	
List all methods of a python object	<code>dir(obj)</code>
How to check the type of one object?	Use type function, e.g. <code>type(enumerate([1, 2, 3]))</code>

1.7 Common Errors

Name	Comment
Error: <code>i++</code>	OK: <code>i += 1</code>
Error: <code>b=true</code>	OK: <code>b=True</code>
Error: <code>i<len(A) && j<len(B):</code>	OK: <code>i<len(A) and j<len(B):</code>
Error: <code>for i>=0 and j>=0:</code>	OK: <code>while i>=0 and j>=0:</code>
Error: <code>! f</code>	OK: <code>not f</code>
NameError: name 'List' is not defined	<code>from typing import List</code>
Python float with high resolution	

1.8 Pip - Python Package Management

Name	Comment
Check on installed python package	<code>pip show simplejson</code>
Search a package	<code>pip search simplejson</code>
Install and uninstall a package	<code>pip install simplejson, pip uninstall simplejson</code>
Install package with a specific version	<code>pip install flake8==2.0</code>
Show installation folder of a module	<code>module.__file__, flask.__file__</code>
Check on-line help for a module	<code>help(module)</code> <code>pip install -U simplejson</code> <code>pip install -i http://pypi.python.jp flask</code>

1.9 Integer

Name	Comment
max, min	<code>sys.maxsize, -sys.maxsize-1</code>
min, max	<code>min(2, 3), max(5, 6, 2)</code>
min with customized comparision	<code>min(a, b, key=lambda x: x*x-2*x+1)</code>
generate range	<code>for num in range(10,20)</code>
get ascii	<code>ord('a'), chr(97)</code>
print integer in binary	<code>"{0:b}".format(10)</code>

1.10 Dict/Hashmap & Set

Name	Comment
dict get first element	<code>m[m.keys()[0]]</code>
get by key with default value	<code>m.get(x, -1)</code>
Dictionary with defaults	<code>m = collections.defaultdict(lambda: 1)</code>
Dictionary with tuple defaults	<code>d=collections.defaultdict(lambda: (0, 0)), d[0, 1]=(2, 3)</code>
Use array as key in dictionary	Convert array to tuple: <code>m[tuple(1)]=3</code>
Check whether key in hashmap	<code>if k in m</code>
Loop dictionary by keys	<code>for k in m</code>
Loop dictionary	<code>for k,v in m.items(), not for k,v in enumerate(m)</code>
Intersection of two sets	<code>list(set(l1).intersection(set(l2)))</code>
List to set	<code>set(list1)</code>
Remove from set	<code>s.remove(2)</code>
Deep copy dict	<code>import copy; m2=copy.deepcopy(m1)</code>
Remove the first from set	<code>s.pop()</code>
Sort dict by values	<code>sorted(dict1, key=dict1.get)</code>
Convert a str to a dict	<code>eval("{\"createTime\":\"2013-07-16\""}")</code>
Delete an element from a dict	<code>del d[key]</code>

1.11 Bit Operator

Name	Comment
mod	<code>x % 2</code>
shift left	<code>x << 1; a << 2</code>
shift right	<code>x >> 2</code>
and	<code>x & y</code>
complement	<code>~x</code>
xor	<code>x ^ y</code>
power	<code>2 ** 3</code>
bool complement	<code>not x</code>
binary format	<code>bin(5) (get 101)</code>
count 1 inside binary	<code>bin(5).count('1')</code>

1.12 File

Name	Comment
Append file	<code>open("/tmp/test.txt", "ab").write("\ntest:")</code>
Write file	<code>open("/tmp/test.txt", "wab").write("\ntest:")</code>
Read files	<code>f.readlines()</code>
Check file	<code>os.path.exists("/tmp/test.txt")</code>
Reference	Github: cheatsheet-python-A4/code/exampleFile.py

1.13 Math

Name	Comment
sqrt	<code>import math; math.sqrt(5)</code>
power	<code>import math; math.pow(2, 3)</code>
log	<code>import math; math.log(5, 2), log2(5)</code>
random	<code>random.randint(1, 10)</code> 1 and 10 included
eval string	<code>eval("2-11*2")</code>

1.14 Networking

Name	Comment
Send http REST call	<code>pip install requests; r = requests.get('https://XX/XX', auth=('user', 'pass'))</code>
Start a simple HTTP server	<code>python -m SimpleHTTPServer <port_number></code>

1.15 Python Interoperate

Name	Comment
Run shell command	<code>output = subprocess.run(["ls", "-lt", "tmp"], stdout=subprocess.PIPE)</code>
Get shell command output	<code>output.stdout.decode('utf-8').split('\n')</code>
Get shell return code	<code>output.returncode, output.check_returncode()</code>
Python trap linux signal	Github: cheatsheet-python-A4/code/exampleSignal.py

1.16 Queue/heapq

Name	Comment
Initialize min heap	<code>heapq.heapify(q)</code>
heappush a tuple	<code>q=[]; heapq.heappush(q, (5, 'ab'))</code>
pop	<code>print (heapq.heappop(q))</code>
first item	<code>q[0]</code>
print heapq	<code>print list(q)</code>
create a queue	<code>from collections import deque; queue = deque([1,5,8,9])</code>
append queue	<code>queue.append(7)</code>
pop queue from head	<code>element = queue.popleft()</code>
Reference	Link: Python Heapq

1.16.1 minheap & maxheap

```
import heapq

# initializing list
li = [5, 7, 9, 1, 3]

# using heapify to convert list into heap
heapq.heapify(li) # a minheap
heapq._heapify_max(li) # for a maxheap!

# printing created heap
print (list(li))

# using heappush() to push elements into heap
# pushes 4
```

```

heapq.heappush(li,4)

# printing modified heap
print (list(li))

# using heappop() to pop smallest element
print (heapq.heappop(li))

print (list(li))

```

1.17 Code snippets

- Initialize Linkedlist from array

```

def initListNodeFromArray(self, nums):
    head = ListNode(None)
    prev, p = head, head
    for num in nums:
        pre = p
        p.val = num
        q = ListNode(None)
        p.next = q
        p = p.next
    pre.next = None
    return head

```

- Print linkedlist

```

def printListNode(self, head):
    print("printListnode")
    while head:
        print("%d" % (head.val))
        head = head.next

```

- Print Trie Tree in level order

```

def printTrieTreeLevelOrder(self, node):
    print("printTrieTreeLevelOrder")
    if node.is_word:
        print("Node is a word")
    queue = []
    queue.append(node)
    while len(queue) != 0:
        s = ''
        for i in range(len(queue)):
            node = queue[0]
            del queue[0]
            for child_key in node.children:
                s = '%s %s' % (s, child_key)
                queue.append(node.children[child_key])
        if s != '':
            print 'print level children: %s' % (s)

```

- python sort with customized cmp function: -1 first

```

nums = [3, 2, 6]
def myCompare(v1, v2):
    return -1
sorted_nums = sorted(nums, cmp=myCompare)
print nums # [3, 2, 6]
print sorted_nums # [6, 3, 2]

```

- Initialize m*n matrix

```
col_count, row_count = 3, 2
matrix = [[None for j in range(col_count)] for i in range(row_count)]
print matrix
```

1.18 Python Common Algorithms

Num	Category/Tag	Example
1	#bfs	LeetCode: Max Area of Island
2	#dfs	LeetCode: Surrounded Regions
3	#binarysearch	LeetCode: Search Insert Position
4	#interval, #mergelist	LeetCode: Interval List Intersections
5	#twopointer, #array	LeetCode: Reverse Words in a String II
6	#twopointer	LeetCode: Two Sum
7	#backtracking, #subset	LeetCode: Subsets II
8	#linkedlist, #presum	LeetCode: Remove Zero Sum Consecutive Nodes from Linked List
9	#unionfind	LeetCode: Accounts Merge
10	#trie	LeetCode: Longest Word in Dictionary
11	#stack	LeetCode: Valid Parentheses
12	#stack	LeetCode: Reverse Substrings Between Each Pair of Parentheses
13	#heap	LeetCode: Top K Frequent Elements
14	#baseconversion	LeetCode: Base 7, LeetCode: Convert to Base -2
15	#interval	LeetCode: Meeting Rooms II, LeetCode: My Calendar I
16	#monotone	LeetCode: Daily Temperatures
17	#knapsack	LeetCode: Coin Change
18	#sortbyfunction	LeetCode: Relative Sort Array
19	#slidingwindow	LeetCode: Longest Substring Without Repeating Characters
20	#editdistance, #dynamicprogramming	LeetCode: Longest Common Subsequence
21	#twopointer, #mergetwolist	LeetCode: Merge Sorted Array
22	#topologicalsort	LeetCode: Course Schedule
23	#bfs, bidirectional bfs	LeetCode: Word Ladder
24	#monotonicfunc, #binarysearch	LeetCode: Kth Smallest Number in Multiplication Table
25	#divideconquer, #recursive	LeetCode: Count of Smaller Numbers After Self
26	python semaphore	LeetCode: Print Zero Even Odd

1.19 More Resources

License: Code is licensed under MIT License.

https://www.tutorialspoint.com/python_data_structure/index.htm