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

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 libaries/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	`print(callable(1))`, `print(callable(lambda:  1))`
Python long	
Python Constants vs Literals	
How functools.$\text{lru}_{\text{cache}}$ works	
Python yield	
Reference	Link: Python Design and History FAQ

1.3 List & Tuples

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

1.4 String

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

1.5 Stack & Queue

Name	Comment
Python deque as a stack	s = collections.deque(), s.append(x), s.pop(), s[-1]
Python deque as a queue	s = collections.deque(), s.append(x), s.popleft(), s[0]
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	`add-apt-repository ppa:deadsnakes/ppa`, `apt install python3.7`
Python constants	
Python nested function	Github: cheatsheet-python-A4/code/nestedFunction.py
Run python snippet from shell	python -c 'import sys; print(sys.getdefaultencoding())'
What's Python Literals	
List all methods of a python object	`dir(obj)`
How to check the type of one object?	Use type function, e.g, `type(enumerate([1, 2, 3]))`

1.7 Common Errors

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

1.8 Pip - Python Package Management

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

1.9 Integer

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

1.10 Dict/Hashmap & Set

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

1.11 Bit Operator

Name	Comment
mod	`x % 2`
shift left	`x « 1; a « 2`
shift righ	`x » 2`
and	`x & y`
complement	`~x`
xor	`x ^ y`
power	`2 ** 3`
bool complement	`not x`
binary format	`bin(5)` (get 101)
count 1 inside binary	`bin(5).count('1')`

1.12 File

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

1.13 Math

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

1.14 Networking

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

1.15 Python Interoperate

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

1.16 Queue/heapq

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

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