总结了30段极简的Python代码！

点击关注 法纳斯特 2022-04-03 14:00

来源丨网络

学 Python 怎样才最快，当然是实战各种小项目，只有自己去想与写，才记得住规则。今天给大家分享的是 30 个极简任务，初学者可以尝试着自己实现；本文同样也是 30 段代码，Python 开发者也可以看看是不是有没想到的用法。

1. 重复元素判定

以下方法可以检查给定列表是不是存在重复元素，它会使用 set() 函数来移除所有重复元素。

```
def all_unique(lst):
    return len(lst) == len(set(lst))
x = [1,1,2,2,3,2,3,4,5,6]
y = [1,2,3,4,5]
all_unique(x) # False
all_unique(y) # True
```

2. 字符元素组成判定

检查两个字符串的组成元素是不是一样的。

```
from collections import Counter

def anagram(first, second):
    return Counter(first) == Counter(second)
anagram("abcd3", "3acdb") # True
```

3. 内存占用

下面的代码块可以检查变量 variable 所占用的内存。

```
import sys
variable = 30
print(sys.getsizeof(variable)) # 24
```

4. 字节占用

下面的代码块可以检查字符串占用的字节数。

```
def byte_size(string):
    return(len(string.encode('utf-8')))
```

```
byte_size('😀') # 4
byte_size('Hello World') # 11
```

5. 打印 N 次字符串

该代码块不需要循环语句就能打印 N 次字符串。

```
n = 2;
s ="Programming";

print(s * n);
# ProgrammingProgramming
```

6. 大写第一个字母

以下代码块会使用 title() 方法，从而大写字符串中每一个单词的首字母。

```
s = "programming is awesome"

print(s.title())
# Programming Is Awesome
```

7. 分块

给定具体的大小，定义一个函数以按照这个大小切割列表。

```
from math import ceil

def chunk(lst, size):
    return list(
        map(lambda x: lst[x * size:x * size + size],
            list(range(0, ceil(len(lst) / size)))))

chunk([1,2,3,4,5],2)
# [[1,2],[3,4],5]
```

8. 压缩

这个方法可以将布尔型的值去掉，例如 `(False, None, 0, "")` ，它使用 `filter()` 函数。

```
def compact(lst):
    return list(filter(bool, lst))
compact([0, 1, False, 2, '', 3, 'a', 's', 34])
# [ 1, 2, 3, 'a', 's', 34 ]
```

9. 解包

如下代码段可以将打包好的成对列表解开成两组不同的元组。

```
array = [['a', 'b'], ['c', 'd'], ['e', 'f']]
```

```
transposed = zip(*array)
print(transposed)
# [('a', 'c', 'e'), ('b', 'd', 'f')]
```

10. 链式对比

我们可以在一行代码中使用不同的运算符对比多个不同的元素。

```
a = 3
print( 2 < a < 8) # True
print(1 == a < 2) # False
```

11. 逗号连接

下面的代码可以将列表连接成单个字符串，且每一个元素间的分隔方式设置为了逗号。

```
hobbies = ["basketball", "football", "swimming"]

print("My hobbies are: " + ", ".join(hobbies))
# My hobbies are: basketball, football, swimming
```

12. 元音统计

以下方法将统计字符串中的元音 (‘a’, ‘e’, ‘i’, ‘o’, ‘u’) 的个数，它是通过正则表达式做的。

```
import re

def count_vowels(str):
    return len(len(re.findall(r'[aeiou]', str, re.IGNORECASE)))

count_vowels('foobar') # 3
count_vowels('gym') # 0
```

13. 首字母小写

如下方法将令给定字符串的第一个字符统一为小写。

```
def decapitalize(string):
    return str[:1].lower() + str[1:]

decapitalize('FooBar') # 'fooBar'
decapitalize('FooBar') # 'fooBar'
```

14. 展开列表

该方法将通过递归的方式将列表的嵌套展开为单个列表。

```
def spread(arg):
    ret = []
    for i in arg:
        if isinstance(i, list):
```

```
            ret.extend(i)
        else:
            ret.append(i)
    return ret

def deep_flatten(lst):
    result = []
    result.extend(
        spread(list(map(lambda x: deep_flatten(x) if type(x) == list else x, lst))))
    return result

deep_flatten([1, [2], [[3], 4], 5]) # [1,2,3,4,5]
```

15. 列表的差

该方法将返回第一个列表的元素，其不在第二个列表内。如果同时要反馈第二个列表独有的元素，还需要加一句 `set_b.difference(set_a)` 。

```
def difference(a, b):
    set_a = set(a)
    set_b = set(b)
    comparison = set_a.difference(set_b)
    return list(comparison)

difference([1,2,3], [1,2,4]) # [3]
```

16. 通过函数取差

如下方法首先会应用一个给定的函数，然后再返回应用函数后结果有差别的列表元素。

```
def difference_by(a, b, fn):
    b = set(map(fn, b))
    return [item for item in a if fn(item) not in b]

from math import floor
difference_by([2.1, 1.2], [2.3, 3.4],floor) # [1.2]
difference_by([{ 'x': 2 }, { 'x': 1 }], [{ 'x': 1 }], lambda v : v['x'])
# [ { x: 2 } ]
```

17. 链式函数调用

你可以在一行代码内调用多个函数。

```
def add(a, b):
    return a + b

def subtract(a, b):
    return a - b

a, b = 4, 5
print((subtract if a > b else add)(a, b)) # 9
```

18. 检查重复项

如下代码将检查两个列表是不是有重复项。

```
def has_duplicates(lst):
```

```
    return len(lst) != len(set(lst))

x = [1,2,3,4,5,5]
y = [1,2,3,4,5]
has_duplicates(x) # True
has_duplicates(y) # False
```

19. 合并两个字典

下面的方法将用于合并两个字典。

```
def merge_two_dicts(a, b):
    c = a.copy()   # make a copy of a
    c.update(b)    # modify keys and values of a with the ones from b
    return c

a = { 'x': 1, 'y': 2}
b = { 'y': 3, 'z': 4}
print(merge_two_dicts(a, b))
# {'y': 3, 'x': 1, 'z': 4}
```

在 Python 3.5 或更高版本中，我们也可以用以下方式合并字典：

```
def merge_dictionaries(a, b)
   return {**a, **b}

a = { 'x': 1, 'y': 2}
b = { 'y': 3, 'z': 4}
print(merge_dictionaries(a, b))
# {'y': 3, 'x': 1, 'z': 4}
```

20. 将两个列表转化为字典

如下方法将会把两个列表转化为单个字典。

```
def to_dictionary(keys, values):
    return dict(zip(keys, values))

keys = ["a", "b", "c"]
values = [2, 3, 4]
print(to_dictionary(keys, values))
# {'a': 2, 'c': 4, 'b': 3}
```

21. 使用枚举

我们常用 For 循环来遍历某个列表，同样我们也能枚举列表的索引与值。

```
list = ["a", "b", "c", "d"]
for index, element in enumerate(list):
    print("Value", element, "Index ", index, )

# ('Value', 'a', 'Index ', 0)
# ('Value', 'b', 'Index ', 1)
```

```
#('Value', 'c', 'Index ', 2)
# ('Value', 'd', 'Index ', 3)
```

22. 执行时间

如下代码块可以用来计算执行特定代码所花费的时间。

```
import time

start_time = time.time()

a = 1
b = 2
c = a + b
print(c) #3

end_time = time.time()
total_time = end_time - start_time
print("Time: ", total_time)

# ('Time: ', 1.1205673217773438e-05)
```

23.Try else

我们在使用 try/except 语句的时候也可以加一个 else 子句，如果没有触发错误的话，这个子句就会被运行。

```
try:
    2*3
except TypeError:
    print("An exception was raised")
else:
    print("Thank God, no exceptions were raised.")

#Thank God, no exceptions were raised.
```

24. 元素频率

下面的方法会根据元素频率取列表中最常见的元素。

```
def most_frequent(list):
    return max(set(list), key = list.count)

list = [1,2,1,2,3,2,1,4,2]
most_frequent(list)
```

25. 回文序列

以下方法会检查给定的字符串是不是回文序列，它首先会把所有字母转化为小写，并移除非英文字母符号。最后，它会对比字符串与反向字符串是否相等，相等则表示为回文序列。

```
def palindrome(string):
    from re import sub
    s = sub('[\W_]', '', string.lower())
    return s == s[::-1]

palindrome('taco cat') # True
```

26. 不使用 if-else 的计算子

这一段代码可以不使用条件语句就实现加减乘除、求幂操作，它通过字典这一数据结构实现：

```
import operator
action = {
    "+": operator.add,
    "-": operator.sub,
    "/": operator.truediv,
    "*": operator.mul,
    "**": pow
}
print(action['-'](50, 25)) # 25
```

27.Shuffle

该算法会打乱列表元素的顺序，它主要会通过 Fisher-Yates 算法对新列表进行排序：

```
from copy import deepcopy
from random import randint

def shuffle(lst):
    temp_lst = deepcopy(lst)
    m = len(temp_lst)
    while (m):
        m -= 1
        i = randint(0, m)
        temp_lst[m], temp_lst[i] = temp_lst[i], temp_lst[m]
    return temp_lst


foo = [1,2,3]
shuffle(foo) # [2,3,1] , foo = [1,2,3]
```

28. 展开列表

将列表内的所有元素，包括子列表，都展开成一个列表。

```
def spread(arg):
    ret = []
    for i in arg:
        if isinstance(i, list):
            ret.extend(i)
        else:
            ret.append(i)
    return ret

spread([1,2,3,[4,5,6],[7],8,9]) # [1,2,3,4,5,6,7,8,9]
```

29. 交换值

不需要额外的操作就能交换两个变量的值。

```
def swap(a, b):
```

```
    return b, a

a, b = -1, 14
swap(a, b) # (14, -1)
spread([1,2,3,[4,5,6],[7],8,9]) # [1,2,3,4,5,6,7,8,9]
```

30. 字典默认值

通过 Key 取对应的 Value 值，可以通过以下方式设置默认值。如果 get() 方法没有设置默认值，那么如果遇到不存在的 Key，则会返回 None。

```
d = {'a': 1, 'b': 2}

print(d.get('c', 3)) # 3
```

> 来源：towardsdatascience
> 作者：Fatos Morina
> 编辑：机器之心
> 原文：
> https://towardsdatascience.com/30-helpful-python-snippets-that-you-can-learn-in-30-seconds-or-less-69bb49204172

万水千山总是情，点个 👍 行不行。

推荐阅读

··· END ···

喜欢此内容的人还喜欢

C++17常用新特性(五)---强制省略拷贝或传递未实质化的对象
CPP开发前沿

1w 字的 pandas 核心操作知识大全!
法纳斯特

尝试再造python编译器:龙书重制版
Coding迪斯尼