整理了100个必备的Python函数，建议收藏！

点击关注 法纳斯特 2022-03-30 14:00

来源 | https://sslljy.blog.csdn.net/?type=blog

前言

新手在做写代码的时候容易卡壳，尤其当接触的函数以及其他知识比较多的时候，经常会看完需求之后不知道自己该用什么方法来实现它，实现的逻辑可能你有，但怎么该用什么函数给忘了，这其实就是知识的储备不够，你记不住哪个函数有什么作用，自然一头雾水。

这几天我专门整理了Python常用的一些函数，从最基础的输入输出函数到正则等12个板块的，总共100多个常用函数，方便小伙伴们进行快速地记忆，每天快速过一遍，用的时候再加深一下，慢慢地你就会摆脱写代码卡壳的状况。

虽说自学编程的时候我们强调更多的东西是理解和实际去敲代码，但有些东西你是要必须牢记的，否则你写代码将寸步难行。老手当然已经烂记于心，新手想要快速得心应手开发，记住高频使用的函数就是一个好法子。

1. 基础函数

序号	函数	说明
1	print()	输出
2	input()	输入
3	int()	转整型
4	float()	转浮点型
5	str()	转字符串
6	type()	返回对象的类型
7	isinstance()	返回布尔值（True,False）

案例：将浮点型数值转换为字符串，输出转换后的数据类型

```
f = 30.5
ff = str(f)
print(type(ff))

#输出结果为 class 'str'
```

2. 流程控制

序号	函数	说明
1	if 语句： 执行1	条件判断
2	if 条件： 代码块1 else：代码块2	条件判断
3	while	判断循环
4	for	计数循环
5	range()	范围函数，可控制开始位置、结束位置和步长
6	break	跳出循环
7	continue	跳过本次循环，后面的循环继续执行

案例：根据用户输入的分数判断成绩，低于50分时提示“你的分数低于50分”，5059分时提示“你的分数在60分左右”，大于等于60分为及格，8090分为优秀，大于90分为非常优秀。

```
s = int(input("请输入分数:"))
if 80 >= s >= 60:
    print("及格")
elif 80 < s <= 90:
    print("优秀")
elif 90 < s <= 100:
    print("非常优秀")
else:
    print("不及格")
    if s > 50:
        print("你的分数在60分左右")
    else:
        print("你的分数低于50分")
```

3. 列表

序号	函数	说明
1	append()	向列表中添加对象，并添加到末尾
2	extend(可迭代对象)	将可迭代对象中数据分别添加到列表中，并添加到末尾
3	insert(下标,对象)	向指定下标位置添加对象
4	clear()	清空列表
5	pop()	删除下标指定的元素，如果不加下标则删除最后一个元素
6	remove(对象)	删除指定的对象
7	del	删除变量或指定下表的值
8	copy()	浅拷贝
9	count(对象)	返回对象在列表中出现的次数
10	index(value,开始下标,结束下标)	元素出现的第一次下标位置，也可自定义范围
11	reverse()	原地翻转
12	sort（key=None，reverse=False）	快速排序，默认从小到大排序，key:算法
13	len()	获取列表的长度（元素）

案例：判断6这个数在列表[1,2,2,3,6,4,5,6,8,9,78,564,456]中的位置，并输出其下标。

```
l = [1,2,2,3,6,4,5,6,8,9,78,564,456]
n = l.index(6, 0, 9)
print(n)

#输出结果为  4
```

4. 元组

序号	函数	说明
1	list(元祖)	元祖转换成列表
2	tuple（列表）	列表转换成元祖
3	元祖的函数操作与列表大致相同相同，不赘述	

案例：修改元组

```
#取元组下标在1~4之间的3个数，转换成列表
t = (1,2,3,4,5)
print(t[1:4])
l = list(t)
print(l)
```

```
#在列表下标为2的位置插入1个6
l[2]=6
print(l)
#讲修改后的列表转换成元组并输出
t=tuple(l)
print(t)
```

```
#运行结果为：

(2, 3, 4)
[1, 2, 3, 4, 5]
[1, 2, 6, 4, 5]
(1, 2, 6, 4, 5)
```

5. 字符串

序号	函数	说明
1	capitalize()	把字符串的第一个字符改为大写，后面的小写
2	casefold()	把整个字符串都小写
3	encode()	编码 str–bytes（二进制字符串）
4	decode()	解码
5	count(sub,start,stop)	返回字符（sub）出现的次数，star：开始下标，stop:结束下标
6	find(sub,start,stop)	返回sub第一次出现的下标,查不到返回-1
7	index(sub,start,stop)	返回sub第一次出现的下标
8	upper()	将字符串转为大写
9	lower()	将字符串转为小写
10	format （）	将字符串按某种格式输出

案例：用format（）的三种方式输出字符串

方式1：用数字占位（下标）

```
"{0} 嘿嘿".format("Python")
a=100
s = "{0}{1}{2} 嘿嘿"
s2 = s.format(a,"JAVA","C++")
print(s2)

#运行结果为：100JAVAC++ 嘿嘿
```

方式2：用{} 占位

```
a=100
s = "{}{}{} 嘿嘿"
s2 = s.format(a,"JAVA","C++","C# ")
print(s2)

#运行结果为：100JAVAC++ 嘿嘿
```

方式3：用字母占位

```
s = "{a}{b}{c} 嘿嘿"
s2 = s.format(b="JAVA",a="C++",c="C# ")
print(s2)
```

```
#运行结果为：C++JAVAC#  嘿嘿
```

6. 字典

序号	函数	说明
1	clear()	清空字典
2	copy()	浅拷贝
3	fromkeys(可迭代对象，value=None)	根据可迭代对象中的元素去创建字典
4	get(key,[d])	获取键所对应的值，key是键，d是提示信息
5	items()	将字典中的键值对封装成元组并放到类集合中
6	pop(key,[d])	根据键删除字典中的键值对，key是键，d是提示信息
7	values()	返回字典中的值（类集合对象）

案例：在字典中查找数据

```
d = {"name": "小黑"}
print(d.get("name2", "没有查到"))
print(d.get("name"))
```

```
#运行结果为：
没有查到
小黑
```

7. 函数

函数这块重头戏更多的是自定义函数，常用的内置函数不是很多，主要有以下几个：

序号	函数	说明
1	函数名.**doc**	获取函数的文档内容
2	help(函数名)	查看函数文档
3	global 变量	声明变量为全局变量（可用于任何地方）
4	nonlocal 变量	声明的变量为全局变量（用于函数嵌套，变量存在于上一级函数）

案例：在函数中定义一个局部变量，跳出函数仍能调用该变量

```
def fun1():
    global b
    b=100
    print(b)
fun1()
print(b)
```

```
#运行结果为：
100
100
```

8. 进程和线程

序号	函数	说明
1	os.getpid()	获取当前进程的编号

2	multiprocessing.current_process()	获取当前进程的名字
3	os.getppid()	获取当前父进程的编号
4	Thread(target=None,name=None,args=(),kwargs=None)	target:可执行目标，name：线程的名字默认Thread-N，args/kwargs:目标参数
5	start()	启动子线程
6	threading.current_thread()	获取当前进程的名字

案例：继承Thread类实现

```
#多线程的创建
class MyThread(threading.Thread):
    def __init__(self,name):
        super().__init__()
        self.name = name
    def run(self):
        #线程要做的事情
        for i in range(5):
            print(self.name)
            time.sleep(0.2)
 #实例化子线程
t1 = MyThread("凉凉")
t2 = MyThread("最亲的人")

t1.start()
t2.start()
```

9. 模块与包

序号	函数	说明
1	import 模块名	导入模块
2	from 模块名 import 功能1，功能2...	导入模块特定功能
3	from 模块名 import *	导入模块所有功能
4	import 模块名 as 别名	模块定义别名
5	from 模块名 import 功能 as 别名	功能定义别名
6	import 包名.模块名.目标	包的导入方式1
7	import 包名.子包名.模块名.目标	包的导入方式2
8	import 包名.模块名	包的使用形式1
9	import 包名.模块名 as 别名	包的使用形式2
10	from 包名.模块名 import 功能	包的使用形式3
11	from 包名 import 模块名	包的使用形式4
12	from 包名.模块名 import *	包的使用形式5

案例：包的使用方式4

```
from my_package1 import my_module3
print(my_module3.a)
my_module3.fun4()
```

10. 文件操作

（1）常规文件操作

序号	函数	说明
1	open(name,mode)	用于打开一个文件，返回一个文件对象 name:文件名，-写全（文件路径+文件名+后缀） mode：打开文件的方式，默认是r--只读
2	write("xxx")	向文件中写入内容
3	read()	读取文件中的内容

4	close()	关闭文件

关于文件操作的常规模式：

模式	描述
r	以只读的形式打开文件，文件的指针在开头
r+	读写，文件指针在开头
rb	以二进制的形式只读文件指针在开头
w	只写，文件不存在，则创建新的，存在则覆盖，指针在开头
w+	读写，文件不存在，则创建新的，存在则覆盖，指针在开头
wb	只写，以二进制的形式
a	追加模式，文件指针在结尾
a+	读写，不存在则创建，存在直接追加
ab	以二进制形式追加

file的对象属性

序号	方法	说明
1	closed	如果文件对象已关闭，返回True，否则返回False
2	mode	返回文件对象的访问模式
3	name	返回文件的名称

file对象的方法

序号	函数	方法
1	close()	关闭文件
2	read([count])	读取文件中的内容，count：字节数量
3	readlines()	读取所有内容，打包成列表
4	readline()	读取一行数据，追加读取，读取过得不能再次读取
5	seek（offset，[from])	修改指针的位置：从from位置移动了offset个字节 from：0-从起始位置，1-从当前位置开始，2-从末尾开始 soofset：要移动的字节数
6	write()	向文件中写入内容

（2）OS模块

- 关于文件的功能

序号	方法	说明
1	os.rename(原文件名,新的文件名)	文件重命名
2	os.remove(文件名)	删除文件

- 关于文件夹的功能

序号	函数	说明
1	mkdir(文件夹名字)	创建文件夹
2	rmdir(文件夹名字)	删除文件夹
3	getcwd()	获取当前目录
4	chdir（目录)	切换目录
5	listdir()	获取当前文件夹下所有文件或文件夹，返回一个列表 listdir('aa') #获取aa文件下所有文件或文件夹，返回一个列表

11. 修饰器/装饰器

序号	函数	说明
1	property	将方法变为属性，被修饰的方法名必须和property下方的方法名一样
2	staticmethod	静态方法，将被修饰的方法从类中抽离出来。该函数不能访问类的属性
3	classmethod	与实例方法的区别是接收的第一个参数不是self，而是cls（当前类的具体类型） 被修饰的方法无法访问实例属性，但是可以访问类属性

案例：classmethod的用法举例

```
class B:
    age = 10
    def __init__(self,name):
        self.name = name
    @classmethod
    def eat(cls): #普通函数
        print(cls.age)

    def sleep(self):
        print(self)

b = B("小贱人")
b.eat()

#运行结果为:10
```

12. 正则

序号	函数	说明
1	re.compile(正则表达式)	编译正则
2	match()	决定re是否在字符串刚开始的位置（匹配行首）
3	search()	扫描字符串，找到这个re匹配的位置（仅仅是第一个查到的）
4	findall()	找到re匹配的所有字符串，返回一个列表
5	group()	返回re匹配的字符串
6	start()	返回匹配开始的位置
7	end()	返回匹配结束的位置
8	span()	返回一个元组：（开始,结束）的位置
9	findall()	根据正则表达式返回匹配到的所有字符串
10	sub(正则,新字符串,原字符串)	替换字符串
11	subn(正则,新字符串,原字符串)	替换字符串，并返回替换的次数
12	split()	分割字符串

案例：用split()函数分割一个字符串并转换成列表

```
import re
s = "abcabcacc"
l = re.split("b",s)
print(l)

#运行结果为: ['a', 'ca', 'cacc']
```

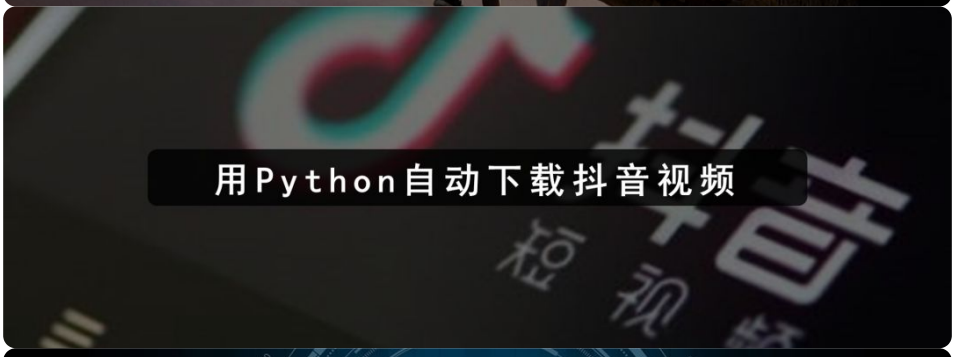
结语

这篇文章的目的，不是为了教大家怎么使用函数，而是为了快速、便捷地记住常用的函数名，所以没有把每个函数的用法都给大家举例，你只有记住了函数名字和它的作用之后，你才会有头绪，至于函数的用法，百度一下就出来，用了几次你就会了。

如果连函数名和它的用途都不知道，你要花的时间和精力就更多了，必然不如我们带着目的性地去查资料会更快些。

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

分享 63 个面向前端开发人员的开源项目工具
前端达人

Stream流递归实现遍历树形结构
知识追寻者

StarRocks的SQL指纹应用
雷雷DBA