Lesson6

Monkey by 51reboot

大纲

- 回顾
- 字符串
- is/==区别
- 赋值/浅拷贝/深拷贝
- 变量生命周期
- 变量的作用域
- 闭包/装饰器
- 可迭代对象、迭代器、生成器
- 多线程/多进程
- Q&A

2

回顾

3

字符串

- 常用方法

```
.count
.format
.isdigit
.islower
.isupper
.lower
.upper
.join
.replace
.split
.startswith
.endswith
.strip
```

- format格式化

```
:^  # 剧中对齐
:<  # 靠左对齐
:>  # 靠右对齐
```

4

示例

```
FILENAME = '/etc/passwd'

with open(FILENAME) as fd:
    for line in fd:
        lineArr = line.strip('\n').split(':')
        # print("{} {} {} {}".format(lineArr[0], lineArr[2], lineArr[5], lineArr[6]))
        print("{:<10} {:^5} {:>30} {:>30}".format(lineArr[0], lineArr[2], lineArr[5], lineArr[6]))
```

5

is/==区别

- 示例
- 总结

6

示例

```
a = 1
b = 1
a == b # True
a is b # True

a = 888
b = 888
a == b # True
a is b # False

a = 'hello'
b = 'hello'
a is b # True
a == b # True

a = 'hello world'
b = 'hello world'
a == b # True
a is b # False
```

7

总结

1. is比较的是两个对象的内存地址是否相等，==比较的是两个对象的值是否相等。
2. python的小整数池和垃圾回收机制了。
   python为了让运行速度快些，在内存中专门开辟了一块区域放置-5到256，
   所有代表-5到256的对象都会指向这个区域。
3. Python解释器中使用了intern（字符串驻留）的技术来提高字符串效率，
   什么是intern机制？即值同样的字符串对象仅仅会保存一份，放在一个字符串储蓄池中，
   是共用的，当然，肯定不能改变，这也决定了字符串必须是不可变对象。
   同时，如果字符串中有空格，默认不启用intern机制。

8

赋值/浅拷贝/深拷贝

- 一文胜千言

```
https://blog.csdn.net/colourful_sky/article/details/81263998
```

9

变量生命周期

- 什么是生命周期
- 全局/局部的生命周期

10

什么是生命周期

- 所谓生命周期就是变量从被创建到被系统回收的过程

11

全局/局部的生命周期

- 全局变量生命周期

1．程序运行时开始创建，程序退出时销毁

- 局部变量的生命周期

1．局部变量在函数执行时才被创建，函数执行结束后局部变量被系统回收
2．局部变量在生命周期内，可以用来存储函数内部临时使用的数据

12

变量的作用域

- 介绍
- 分类
- LEGB法则
- 示例
- global/nolocal
- 总结

13

介绍

在Python程序中创建、改变、查找变量名时，都是在一个保存变量名的空间中进行，我们称之为命名空间，也被称之为作用域。

14

分类

- 全局（global）：在函数外部定义
- 局部（local）：在函数内部定义

15

LEGB法则

- L （Local） 局部作用域
- E （Enclosing function locale） 嵌套作用域
- G （Global module）全局作用域
- B （Buildin）内置作用域

总结：
1. 搜索变量名的优先级：局部作用域 > 嵌套作用域 > 全局作用域 > 内置作用域
2. 变量由内到外，先找自己作用域的变量，然后上级寻找，仍未找到，则报错NameError。

16

示例

```
ok = True
if ok:
    url = "www.51reboot.com"
print(url)
for i in range(10):
    age = i
print(age)
```

17

global/nolocal

```
https://zhuanlan.zhihu.com/p/32050475
```

18

总结

19

示例

```
cnt = 1
def change1():
    tmp_cnt = cnt + 1
    print(tmp_cnt)

def change2():
    cnt = cnt + 1
    print(cnt)
change1()
change2()
```

- 总结

1. def、class、lambda 改变变量作用域的代码段;
2. 在if-elif-else、for-else、while、try-except\try-finally等关键字的语句块中并不会产成作用域

20

闭包/装饰器

- 闭包
- 装饰器

21

闭包概念

- 在一个外函数中定义了一个内函数，内函数里运用了外函数的临时变量，并且外函数的返回值是内函数的引用。

22

装饰器

- 是什么？link
- 解决了什么问题？
- 语法

decorate (https://github.com/467754239/python/blob/master/basic/decorate.md)

23

可迭代对象、迭代器、生成器

24

可迭代对象

- 定义

凡是返回一个迭代器的对象都可以称为可迭代对象；
它并不是指某种具体的数据类型。

- 语法

```
>>> iter?
Docstring:
iter(iterable) -> iterator
```

- 列举常见有哪些？
- fd？
- 结论：

1. 实现了__iter__方法。

迭代器

- 定义

1. 任何`同时`实现了__iter__和__next__方法的对象都是迭代器
2. __iter__返回迭代器自身，__next__返回容器中的下一个值，如果容器中没有更多元素了，则抛出StopIteration异常。

注意：

迭代器是一种带状态的的对象。

26

练习

- range函数是可迭代对象 还是 迭代器？

27

生成器

- 生成器是Python最吸引人的特性之一；
- 生成器是一种特殊的迭代器；
- 生成器 和 迭代器的区别？

1. 生成器一定是迭代器， 但迭代器不一定是生成器；
2. 迭代器要同时实现__iter__ 和 __next__ 方法， 但生成器只需要一个yield关键字即可；

28

多线程/多进程

实战(https://github.com/467754239/python/tree/master/threads)

29

Q&A

zhengyscn by 51reboot (https://www.51reboot.com/)

30

Thank you

zhengyscn@gmail.com (mailto:zhengyscn@gmail.com)
http://github.com/zhengyscn (http://github.com/zhengyscn)