day19：数值转换函数重载，迭代器（高级）

笔记本： Python基础
创建时间： 2018/7/3 17:27 **更新时间：** 2018/7/4 17:17
作者： liuchang_0412@163.com

1.数值转换函数重载

```
str(obj)      #__str__
complex       # __complex__
int(obj)      #__int__
float(obj)    #__float__
bool(obj)     #__bool__
```

示例：

见int.py

2.bool运算符的重载：

格式：

```
def __bool__(self):
    pass
```

作用：

用于if语句的真值表达式中

用于while语句的真值表达式中

用于bool(obj)函数取值

说明：

当没有__bool__方法时，真值测试将以__len__(self)方法的返回值来进行测试布尔值，

示例：

见bool.py

3.in / not in 运算符重载：

格式：

```
def __contains__(self, e):
    pass
```

示例：

in_even.py

#练习：

定义一个表示素数类：

```
class Primes:
    def __init__(self, end):
        """end 用于表示素数的终点
           素数的起始点是2（包含2）
        """
        pass
```

用此类创建一个对象：

```
p1 = Primes(100)
if 50 in p1:
    print("50是素数")
else:
    print("50不是素数")
```

4.索引和切片运算符的重载：

重载方法：

```
__getitem__(self, i)        #用索引/切片获取值
__setitem__(self, i, v)     #设置索引或切片的值
__delitem__(self, i)        #进行删除索引操作
```

作用：

让自定义的对象能进行索引和切片操作

示例：

```
L = [1, 2, 3, 4]
L[2] = 3.14    #赋值
print( L2 )    #索引取值
del L[2]       #删除索引内容
```

#练习：

将index_slice2.py 修改方法：

__setitem__(self, index, value)

myl[::2] = 10   #MyList([10, 2, 10 ,4 ,10)]

函数调用（模拟）重载 （__call__方法）：

作用：

让一个对象能向函数一样被调用

方法格式：

```
def __call__(self, 参数列表):
    pass
```

注：

此重载方法的参数可以使一个或多个( )

5.迭代器（高级）：

①迭代器协议：是指对象（实例）能够使用next(it)函数获取下一项数据，
在没有下一项数据时，触发一个StopIteration异常来终止迭代的约定

②next(it)：

对应 __next__(self)方法

③iter(obj)：

对应 __iter__(self)方法，通常返回一个可迭代的对象

④for语句和推导式，先调用iter(obj)拿出迭代器，再迭代

示例：
见odd.py

6. with 语句管理文件：
①语法：
with 表达式 [as 变量名]：
pass
②说明：
as 子句中的变量绑定生成的对象
③示例：
with.py
④作用：
使用于对资源进行访问的场合，确保使用过程中不管是否发生异常，
都会执行必须的“清理”工作，并释放资源

7.环境管理器：
1.类内有__enter__方法 和 __exit__方法的类，被称为环境管理器
2.能够用with语句进行管理的对象必须是环境管理器
3.__enter__将在进入with语句时被调用并返回，由as变量管理对象
4.__exit__将在离开with时被调用，切可以用参数来判断在离开with语句时是否有异常发生，并作出相应的处理

示例：
见cooker.py

#练习：
1.迭代器协议练习：
写一个实现迭代器协议的类，让此类可以生成从b开始到e结束的全部素数。

```
class Primes:
    def __init__(self, b ,e):
        pass

L = x for x in Primes(10, 20)
print(L)            #[11, 13, 17 ,19]
prms = Primes(40, 100)
for x in range(30, 50)
    if x in prms:
        print("x是prms里的素数")
```

#思路：
__init__
__del__

__next__
__iter__
__contains__
is_prime()

2.写一个实现迭代器协议的类，让此类可以生成前n个斐波那契数：

```
class Fibonacci:
    def __init__(self, n):
        pass

for x in Fibonacci(8):
    print(x)    #1 1 2 3 5 ...
print("前10个斐波那契数的和为：", sum(Fibonacci(10)))
```

#思路：
__init__
__del__
__next__
__iter__
__sum__

/c/Users/28952/.ssh/id_rsa
https://github.com/liuchang0412/Python-Study