EDU

CSDN学院　IT实战派

程序设计与数据结构

常用设计模式

讲师：周宇航

课程目标

1. 列表生成式
2. 生成器
3. 迭代器

列表生成式

- Python内置的非常简单却强大的可以用来创建list的生成式
- 快速的把字典内容转变成list

```
>>> d = {'x': 'A', 'y': 'B', 'z': 'C' }
>>> [k + '=' + v for k, v in d.items()]
['y=B', 'x=A', 'z=C']
```

生成器

- 循环的过程中不断推算出后续的元素呢？这样就不必创建完整的list，这种一边循环一边计算的机制，称为生成器：generator
- 两种定义
    1. (列表生成式)
    2. 带yield的generator function

```
L = [x * x for x in range(10)]
print(L)
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
g = (x * x for x in range(10))
print(L)
<generator object <genexpr> at 0x1034dc667>
```

迭代器

- for循环可以应用下列类型：
  1. 集合数据类型，list，tuple，dict，set，str等
  2. generator ，生成器和带yield的generator function等

- 直接可作用于for循环的叫可迭代对象：Iterable

- 直接可作用于next方法的叫可以生成器对象：Iterator

- 生成器可以同时作用于for循环和next（）函数，不断调用，直到抛出StopIteration错误

小结

- 使用列表生成式生成list，更简单，代码量少
- 生成器两种定义方式，惰性序列，调用next方法才会计算
- 生成器对象可for可next，迭代器只可for
- list str dict等可迭代对象，可以转化成生成器对象

布置作业

```
# 输出:
# [1]
# [1, 1]
# [1, 2, 1]
# [1, 3, 3, 1]
# [1, 4, 6, 4, 1]
n = 0
results = []
for t in triangles():
    print(t)
    results.append(t)
    n = n + 1
    if n == 10:
        break
if results == [
    [1],
    [1, 1],
    [1, 2, 1],
    [1, 3, 3, 1],
    [1, 4, 6, 4, 1],
]:
    print('测试通过!')
else:
    print('测试失败!')
```

- 注意：一定要自己动手练习，此作业不用上交

实现triangles生成器

EDU

CSDN学院　IT实战派