无限迭代器：计数器（itertools.count()）

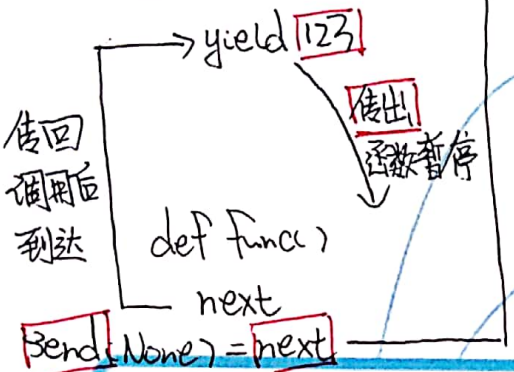
有限迭代器：itertools.chain()

RuntimeError：字典进行插入操作后，字典迭代器会立即失效

尾插入操作不会损坏指向当前元素的List迭代器，列表会自动变长

yield 可以返回值但无法接收值

x = yield　x 接收值



Ansible工具（管理肥来）

优化运行效率

- 多进程
- 多线程
  - I/O密集型
  - 网络爬虫
- 协程
  - 概念
    - 并行和并发
    - 异步编程 事件循环、运行暂停、回调函数
    - 线程和协程的关系
      - 抢占式 非抢占式
      - 被动调度 主动调度
  - 迭代器
    - 迭代器协议
      - for in
      - __next__
      - __iter__
      - __getitem__
      - __next__
    - 分类
      - 无限迭代器
      - 有限迭代器
  - yield 表达式
    - Python3.3引入 yield from
    - ①替代内层循环
    - ②返回给调用方yield执行结果
  - next()调用
  - Python3.5增加 await、async
  - await对象
    - coroutine
    - task
    - future
    - loop
    - awaitable
  - aiohttp
    - 客户端 requests
    - 服务端 flask
    - from aiohttp import web

async def py35_func():
　await sth()

异步

I/O轻量级

asyncio.run() 执行

多协程由一个调度器处理

CPU密集型、计算

进程池和协程：

协程调用过程：（事件循环）

调用协程时，会被注册到 ioloop，返回 coroutine 对象

用 ensure_future 封装为 future 对象

提交给 ioloop

数据获取 → 数据整理 → 对比分析 关联关系 分析建模 → 调试

数据聚合 df.groupby('star').sum() count()

创建新列 df['new_star'] = df['star'].map(star_to_number)

多列排序 df.sort_values(by=['col1','col2'], ascending=[T,F])

表的横向连接 pd.merge(data1, data2, on='group', how='inner')

连接键类型，解决没有公共列问题

pd.merge(data1, data2, left_index='a', right_index='b')
指定连接键： on on

连接试：默认 inner 取两表公共部分
left 以左表为基础，右表在左表拼接
right 以右表为基础，左表在右表拼接
outer 取两表并集

纵向拼接：pd.concat([data1, data2]) 相同列

字符串 字典
--str-- --getitem-- --setitem-- --delitem--
--iter-- --call-- --eq-- --gt-- --ge--
迭代器 比较
--add-- --sub-- --hash--

数据预处理 pandas
Python与pandas交互可用魔术方法

聚合：df2.groupby('type').aggregate() 单次多种统的汇总.
({'type':'count', 'feb':'sum'})

data.groupby('group').transform('mean')
取完之后写入到每一个值上面.

数据透视表：~~未做子分组~~ pd.pivot_table

数据导出：df.to_excel(excel_writer=r'file.xlsx', (csv)/to_pickle速度快只有pandas兼容.
sheet_name='sheet1', index=False,
columns=['col1','col2'])

缺失值处理：na_rep=0 inf_rep=0

- 数据集
  - 深度学习数据集 [sklearn]
  - 国家统计局
- 数据结构
  - 取单行 或单列
  - Series (tuple) pd.series
  - 底层 numpy
  - Dataframe
  - 导入不同格式数据在Pandas合并
- 预处理
  - 缺失值 Series: x.hasnans DataFrame: isnull()
  - 重复值 x.drop_duplicates()
  - 索引
    - 行选择 df.loc[0:2]
    - 列选择 df[1:3] 前3行
    - df.columns=['shorts']
- 数据调整
  - 一对一：df.replace(np.NaN,0)
  - 多对一：df.replace([100,200,300],100)
  - 多对多：df.replace({100:10, 200:20, 300:30})
  - 数值替换
  - 数值删除 df.drop(df.columns, axis=1) ['a','b']
  - 新行列插入
  - 索引重塑
  - df.stack() 数据透视表
  - df.stack().reset_index() 填充索引
- 数值运算
  - 基本运算
  - 汇总
  - 数据分组
  - df['new_star'].sum()
  - sum字符串进行连接
- 多表操作
  - 多表拼接
  - 连接键
  - 连接方式
- 数据导出
  - 导出类型
  - 可视化 matplotlib.pyplot
  - 连接数据库 pymysql

```
import seaborn as sns
sns.set_style('darkgrid')
plt.scatter(df.index, df['A'])
plt.show()
```