Python 数据分析

第一章 Python 基础

1.1 安装

> 注意 配置环境变量

1.2 Python 解释器

- Cpython
- Ipython

1.3 第一个Python 程序

```
print('hello world')
a = "aa"
a=1
a=[1,2,3]
```

1.4 输入输出

- 输出-print

```
--1.
name = 'World'
print('hello %s'%(name))
# hello world
--2. str.format() 格式化字符串
print('my name is {name},age is {age}'.format(name='justin',age=18))
-- 3. f-string 格式化字符串 推荐
name = 'World'
print(f'hello,{name}')
-- 4.不换行输出
for i in range(0,4):
    print(i,end='')
```

- 输入-input()

> 参考资料：https://segmentfault.com/a/1190000018081959

1.5 数据类型和变量

- 整数
- 浮点数

```
精准浮点数计算
from decimal import Decimal
a = Decimal('4.2')
b = Decimal('2.1')
print(a+b)
```

> 参考资料：https://blog.csdn.net/weixin_33695082/article/details/88947108

- 字符串
- 布尔值
  - 布尔值可以用and,or 和 not 运算
- 空值 None
- 变量

> 变量类型不固定的语言称为动态语言，与之对应的是静态语言

1.6 list 和 tuple

- list 有序列表

> https://www.runoob.com/python3/python3-list.html

- tuple

> https://www.runoob.com/python3/python3-tuple.html

1.7 条件判断

- if 语句

```
if condition_1:
    statement_block_1
elif condition_2:
    statement_block_2
else:
    statement_block_3
####################################
score = 100
s = 'A' if score>=90 else ('B' if score>=60 else 'C')
print(s)
```

1.8 循环

- for
- while

```
n = 100

sum = 0
counter = 1
while counter <= n:
    sum = sum + counter
    counter += 1

print("1 到 %d 之和为: %d" % (n,sum))
```

> 注意 while 循环可以使用 else 语句

```
count = 0
while count < 5:
   print (count, " 小于 5")
   count = count + 1
else:
   print (count, " 大于或等于 5")
```

- break 可以跳出 for while 的循环体。
- continue 语句用来告诉Python 跳出当前循环块中的剩余语句，然后进行下一轮循环
- pass

1.9 使用dict 和 set

- dict

```
1、遍历key值，value值（下面写法完全等价）：
a = {'a': '1', 'b': '2', 'c': '3'}
方式一：
for key in a:
    print(key+':'+a[key])
方式二：
for key in a.keys():
    print(key+':'+a[key])
方式三：
for key,value in a.items():
       print(key+':'+value)
方式四：
for (key,value) in a.items():
    print(key+':'+value)
打印结果：
a:1
b:2
c:3

2、遍历value值：
for value in a.values():
    print(value)
打印结果：
1
2
3

3、遍历字典项
for kv in a.items():
    print(kv)
打印结果：
('a', '1')
('b', '2')
('c', '3')
```

- set 无序的不重复元素序列

> https://www.runoob.com/python3/python3-set.html

1.10 Python 函数中的参数传递

```
a = 1
def fun(a):
    a = 2
fun(a)
print (a)
############################
a = []
def fun(a):
    a.append(1)
fun(a)
print (a)
```

- 不可更改对象(string,tuple,numbers)
- 可更改对象(list,dict,set)

1.11 函数

- 函数的参数
  - 位置参数

```
def power(x,n):
    s =1
    while n>0:
        n = n-1
        s = s*x
    return s
```

  - 默认参数

```
def power(x,n=2):
   s =1
   while n>0:
       n = n-1
       s = s*x
   return s
```

  - 可变参数

```
def calc(*numbers):
    sum =0
    for n in numbers:
        sum = sum+n
    return sum
################################
calc(*[i for i in range(1,101)])
# 5050
```

  - 关键字参数

```
extra = {'city':'Beijing','job':'Engineer'}
def person(name,age,**kw):
     print('name:', name, 'age:', age, 'other:', kw)
person('Jack', 24, **extra)
#name: Jack age: 24 other: {'city': 'Beijing', 'job': 'Engineer'}
```

> 具体参考：https://www.liaoxuefeng.com/wiki/1016959663602400/1017261630425888

- 递归函数

```
def fact(n):
    if n==1:
        return 1
    return n * fact(n - 1)
##########循环#######
def fact(n):
    result = 1
    for i in range(1,n+1):
        result *=i
    return result
```

1.12 高级特性

- 切片

```
字符串反转：
str1 ='abcdefg'
str1[::-1]
#'gfedcba'
```

- 列表生成式(列表推导式)

```
# 简单用法
list1 = list(range(1,101))
print(list1)
# 一般用法
list2 = [i+1 for i in range(1,101)]
print(list2)
# 高级用法
list3 = [i for i in range(1,101) if i%2 ==0]
```

1.13 函数式编程

- 高阶函数
  - map

```
a,*b = map(int,input().strip().split())
```

  - reduce

```
from functools import reduce
def add(x,y):
    return x+y
reduce(add,[1,3,5,7,9])
```

- filter

```
def not_empty(s):
    return s and s.strip()
list(filter(not_empty,['A', '', 'B', None, 'C', '  ']))
```

- sorted

```
list3 = [-1,2,-100,3,-4,5]
print(sorted(list3,key=abs))#[-1, 2, 3, -4, 5, -100]
list3
# [-1, 2, -100, 3, -4, 5]
```

- 匿名函数

```
list(map(lambda x:x*x,[1,2,3,4,5,6,7,8,9]))
```

1.14 模块

- pip pip3
- Anacona conda install

第二章 常用的数据分析包

2.1NumPy

- 概念

  NumPy 是Python 数值计算的基石，它提供多种数据结构，算法以及大部分涉及Python 数值计算所需的接口

- 特点

  计算快

```
# 普通计算两个数组相乘的耗时
def chengfa(a,b):
    sum =0
    for i in range(len(a)):
        sum += a[i]*b[i]
    return sum
a = [i for i in range(1,10001)]
b = [j for j in range(10001,20001)]
%timeit chengfa(a,b)
#769 µs ± 44.3 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)
############################
# 使用 numpy 查看效率
import numpy as np
```

```
a = np.array(a)
b = np.array(b)
%timeit a*b
#6.95 µs ± 17.4 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)
```

- 方法

  面向数组编程

  > https://zhuanlan.zhihu.com/p/84103289

- 官方文档

  > https://www.numpy.org.cn/user/quickstart.html#%E5%85%88%E5%86%B3%E6%9D%A1%E4%BB%B6

2.2 Pandas

- pandas 官方文档

  > https://www.pypandas.cn/docs/getting_started/dsintro.html#dataframe

- pandas 与 SQL对比

  > https://blog.csdn.net/weixin_39791387/article/details/81391621

- pandas 操作数据库

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
# 导入包
import os
from sqlalchemy import create_engine
import pandas as pd

os.environ['NLS_LANG'] = 'SIMPLIFIED CHINESE_CHINA.UTF8'

# 同样的套路，创建连接引擎
engine = create_engine('oracle+cx_oracle://user:pass@host:port/dbname')

# with管理安全
with engine.connect() as conn, conn.begin():
    # 直接给出要查的表名，sql原生语句都不用写了
    data = pd.read_sql_table('table_name', conn)
print(data.head())  # 查看前5个数据
```

> https://blog.csdn.net/njulpy/article/details/85110633

- Python 执行存储过程

```
import cx_Oracle
import datetime
def call_delete():
    conn_str='XXXX'
    conn = cx_Oracle.connect(conn_str)
    cur = conn.cursor()
    nowYear = str(datetime.datetime,now().year)
    msg = cur.var(cx_Oracle.STRING)
    cur.callproc('proc_name',[nowYear,msg])
    conn.commit()
    print(msg.getvalue())
if __name__ == "__main__":
    call_delete()
```

> https://blog.csdn.net/Unstoppable365/article/details/109092522 + Jupyter 下划线含义

第三章 数据分析实战