

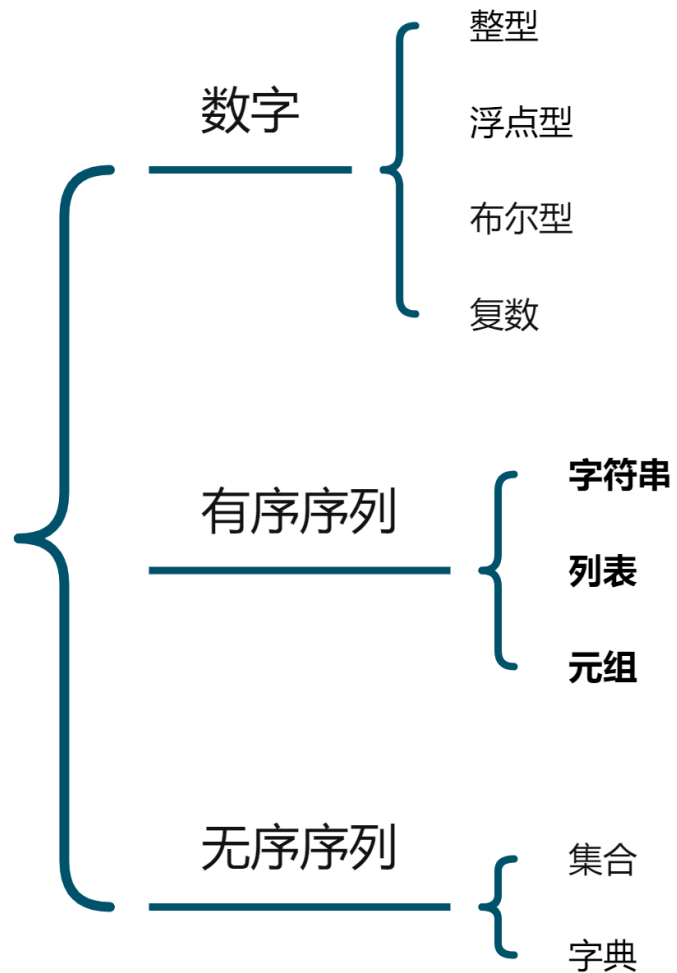
# Python与金融数据挖掘(3)

文欣秀

[wenxinxiu@ecust.edu.cn](mailto:wenxinxiu@ecust.edu.cn)

# 数据类型

Python数据类型



# 单词/词语替换问题

编写程序实现替换功能：从网上拷贝一篇财经短文存入字符串中，实现该短文中某单词/词语的全部替换。

请输入原文：财经新闻属于新闻的一个细分类目，侧重点是采集、报道、发布财经领域的新闻。财经新闻有广义和狭义之分。广义的财经新闻或称泛经济新闻，覆盖全部社会经济生活和与经济有关的领域。狭义的财经新闻，则重点关注资本市场，并用金融资本市场的视角看中国经济主义生活。

被替代字：财经新闻

替代为字：FN

FN属于新闻的一个细分类目，侧重点是采集、报道、发布财经领域的新闻。FN有广义和狭义之分。广义的FN或称泛经济新闻，覆盖全部社会经济生活和与经济有关的领域。狭义的FN，则重点关注资本市场，并用金融资本市场的视角看中国经济主义生活。

# 常用字符串方法

**s. upper():** 将字符串都转换成大写字母

**s. lower():** 将字符串都转换成小写字母

**s. split():** 实现字符串的分割

**s. replace():** 用第二个子串替代第一个子串

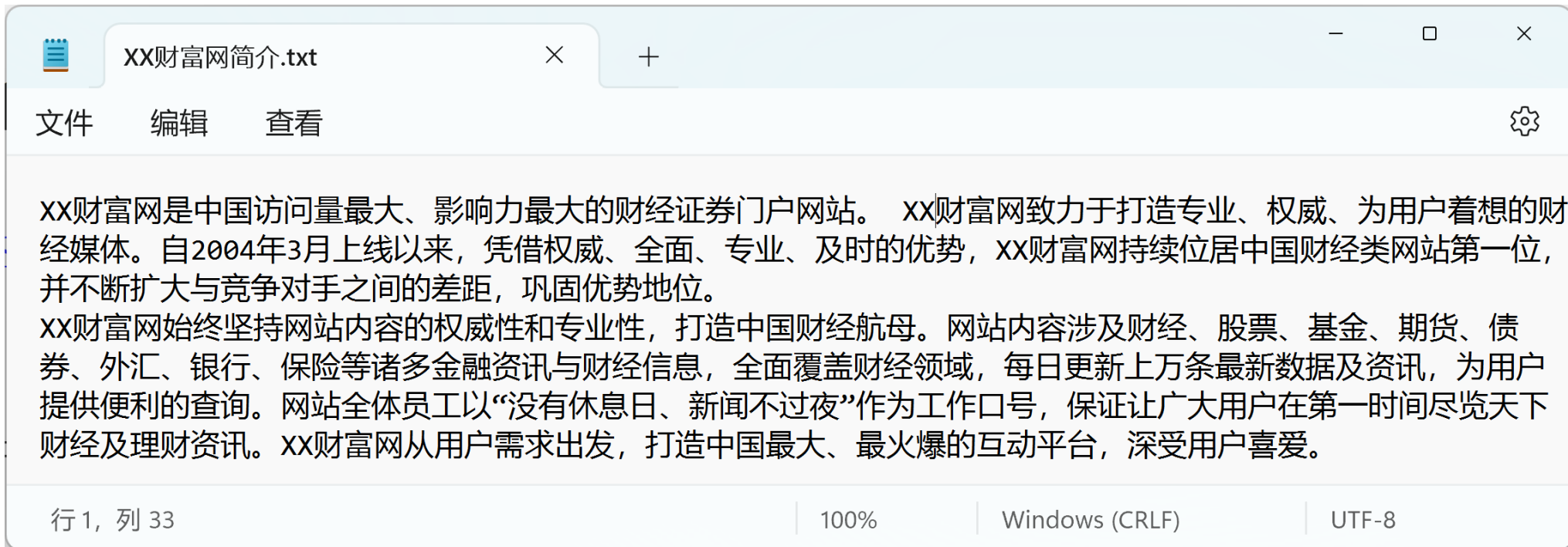
**s. strip():** 消除字符串两端的空格及符号

# 单词/词语替换问题答案

```
paper=input("请输入原文：")  
before=input("被替代字词：")  
after=input("替代为字词：")  
result=paper.replace(before, after)  
print(result)
```

# 拓展问题

编写程序，实现文件“xx财富网简介.txt”中的XX全部替换。



# 文件操作

**文本文件：**基于字符编码。在Python3中，文本文件以字符的Unicode码值进行存储和编码。

**二进制文件：**基于值编码的文件，存储的是二进制数据。

打开文件



读写文件



关闭文件

# 文件操作

- ◆ 文件打开方式一(打开指定位置的文件):

```
handle = open("D:\\exam\\test.txt", 'r')
```

- ◆ 文件打开方式二(打开当前目录下文件):

```
handle = open("test.txt", 'r')
```

- ◆ 文件关闭:

```
handle.close()
```



# 文本文件打开方式

模式	说明
" <b>r</b> "	打开文件以 <b>读取</b> 文件
" <b>w</b> "	打开文件以 <b>写入</b> 文件
" <b>a</b> "	打开文件以 <b>添加</b> 数据
" <b>r+</b> "	打开文件以 <b>读取和写入</b> 数据
" <b>w+</b> "	打开文件以写入和读取数据
" <b>a+</b> "	打开文件以添加和读取数据

# 文件读操作一

**handle.read(n)** : 从文件读取n个字符到字符串

例: test=handle.read(10)

**handle.read()**: 读取整个文件到字符串中

例: test=handle.read()

# 文章单词替换问题

```
fobj=open("XX财富网简介.txt","r",encoding="utf-8")
paper=fobj.read()
fobj.close()
before=input("被替代字: ")
after=input("替代为字: ")
result=paper.replace(before,after)
print(result)
```

# 文章单词替换问题

```
with open("XX财富网简介.txt","r",encoding="utf-8") as fobj:  
    paper=fobj.read()  
    before=input("被替代字: ")  
    after=input("替代为字: ")  
    result=paper.replace(before,after)  
    print(result)
```

# 营收额排序问题

编写程序，自动模拟产生某公司10个月的营收额（1000万-5000万之间的随机数），实现营收额的从小到大排序。

排序前的营收额：

3325 4361 3680 3763 2969 3112 2636 3034 4265 1221

排序后的营收额：

1221 2636 2969 3034 3112 3325 3680 3763 4265 4361

# 列表定义

- ◆能保存任意数量任意类型的Python 对象
- ◆列表元素用中括号 `[]` 包裹，元素用逗号分隔
- ◆第一个元素索引为 `0`，最后一个元素索引为 `-1`
- ◆切片运算符 `[ i : j]` 得到从下标 `i` 到下标 `j-1` 的子集
- ◆元素的个数及元素的值可以改变

# 列表示例

```
>>> aList=[]
```

#定义一个空列表

```
>>> aList=[1, 2, 3, 4]
```

```
>>> aList[-1]
```

#获取最后一个元素

```
>>> aList[2:]
```

#获取一个子列表

```
>>> aList[1]=5
```

#修改一个元素

# 列表函数

>>> **len(L):** 返回列表L的长度，即元素的数量

>>> **max(L):** 返回列表L中的最大元素

>>> **sum(L):** 返回列表L中所有元素(数字)总和

>>> **sorted(L):** 对任意列表L进行排序



# 列表方法

**t.append(x):** 在列表的末尾添加元素，列表的长度增加1

**t.sort(reverse=False):** 将列表中元素进行从小到大排序

**t.remove(x):** 删除列表中的第一个值为x的元素

**t.count(x):** 返回列表中x出现的次数，若不包含x，返回0

# 列表方法示例

```
>>> aList=[1, 20, 5, 8]
```

```
>>> aList. append(17)
```

```
>>> aList. remove(8)
```

```
>>> aList. count(5)
```

```
>>> aList. sort()
```

# 营收额排序问题答案

```
import random
myList=[]
for i in range(10):
    number=random. randint(1000,5000)
    myList. append(number)
print("\n\n排序前的营收额： ")
for i in myList:
    print("{ }".format(i),end=" ")
myList. sort()
print("\n\n排序后的营收额： ")
for i in myList:
    print("{ }".format(i), end=" ")
```

# 拓展问题

编写程序，从文件“price.csv中”读入某公司最近20天的收盘价，输出最高收盘价。

	A
1	181.77
2	128.24
3	136.74
4	167.51
5	127.3
6	131.21
7	140.96
8	114.53
9	154.99
10	120.32

11	141.58
12	102.65
13	175.81
14	125.56
15	198.64
16	145.85
17	127.72
18	134.08
19	178.97
20	167.18

# 文件读操作二

**handle.readline() :** 从文件读一行数据到字符串

例: `handle.readline()`

**handle.readlines() :** 读取整个文件并创建列表

例: `handle.readlines()`

# 读文件示例

	A
1	181.77
2	128.24
3	136.74
4	167.51
5	127.3
6	131.21
7	140.96
8	114.53
9	154.99
10	120.32
11	141.58
12	102.65
13	175.81
14	125.56
15	198.64
16	145.85
17	127.72
18	134.08
19	178.97
20	167.18

```
income=[]
with open("price.csv","r") as fobj:
    money=fobj.readlines()
    for i in money:
        i=i.strip()
        i=float(i)
        income.append(i)
print("最高收盘价为: {:.2f}".format(max(income)))
```

**CSV:** 以逗号分隔的文本文件，xlsx文件可以另存为csv文件

# 文件读操作三

## ◆ 直接在文件对象上循环读取内容

	A
1	181.77
2	128.24
3	136.74
4	167.51
5	127.3
6	131.21
7	140.96
8	114.53
9	154.99
10	120.32
11	141.58
12	102.65
13	175.81
14	125.56
15	198.64
16	145.85
17	127.72
18	134.08
19	178.97
20	167.18

```
income=[]
with open("price.csv","r") as fobj:
    for i in fobj:
        i=i. strip()
        i=float(i)
        income. append(i)
print("最高收盘价为: {:.2f}".format(max(income)))
```

# 股票代码和名称合并问题

已知部分股票代码和股票名称分别存在两个列表中，编写程序，将股票代码和股票名称合并到一个列表中。

代码	名称
000001	上证指数
399001	深证成指
899050	北证50
000300	沪深300
399005	中小100
399006	创业板指

代码	名称
('000001',	'上证指数')
('399001',	'深证成指')
('899050',	'北证50')
('000300',	'沪深300')
('399005',	'中小100')



# 元组定义

- ◆能保存任意数量任意类型的Python 对象
- ◆元组元素用小括号 ( )包裹
- ◆元素的个数及元素的值不可以改变
- ◆索引运算符[ i ]得到下标为i的元素
- ◆切片运算符[ i : j ]得到从下标i到下标j-1的子集

# 元组示例

```
>>> aTuple = ('robots', 77, 93, 'try')
```

```
>>> aTuple[0]
```

```
>>> aTuple[1:3]
```

```
>>> aTuple[::2]
```

# 课堂练习

表达式  $(1, 2) + (3, 4)$  的值为 ( )

A、  $(1, 2, 3, 4)$

B、  $(4, 6)$

C、 10

D、  $(16,)$

# 元组创建方法

>>> t1=()	#创建一个空元组
>>> t2=(1,2,3)	#创建包含三个元素的元组
>>> <b>t3=(1,)</b>	#创建一个包含一个元素的元组
>>> t4=tuple([1,2,3])	#将列表转换为元组
>>> t5=1,2,3	#元组打包
>>> name, age=("Jack",28)	#将元组解包

# 元组应用范围

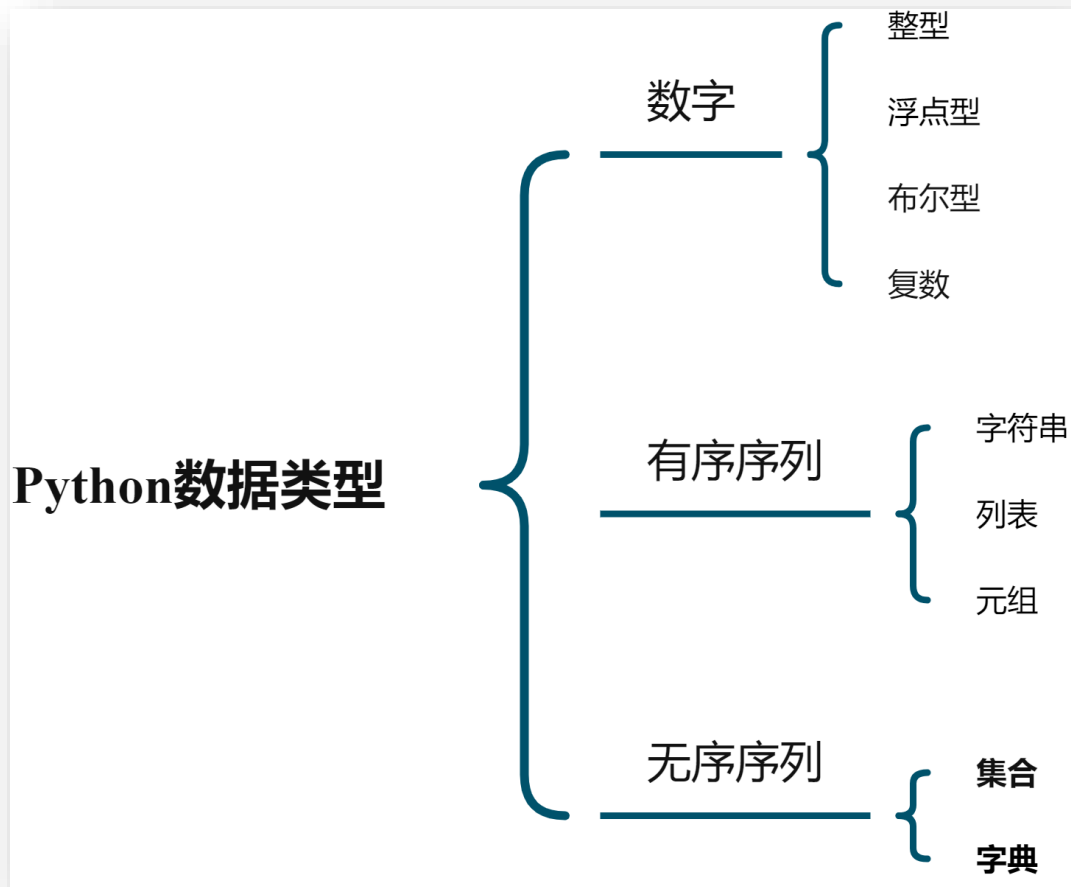
- ◆不能修改、增加或删除元组中的元素
- ◆del 可删除整个元组，但不能删除元素
- ◆对元组的访问和处理速度要快于列表
- ◆可用于函数参数传递，避免参数被修改

# 股票代码和名称合并答案

```
merge=[]
code=["000001","399001","899050","00300","399005"]
name=["上证指数","深证成指","北证50","沪深300","中小100"]
for i in range(len(code)):
    merge.append((code[i],name[i]))
print("  代码      名称")
for e in merge:
    print(e)
```



# 数据类型



代码	名称
000001	上证指数
399001	深证成指
899050	北证50
000300	沪深300
399005	中小100
399006	创业板指



# 字典定义

- ◆是Python 中的映射数据类型， 用{ }包裹
- ◆由**键-值**对构成， 键值对使用冒号:分隔
- ◆键必须唯一， 必须是不可变数据类型
- ◆一般以**数字、字符串、元组**等不可变对象作为键
- ◆值可以是**任意类型**的Python 对象

# 字典示例

>>> test = {} #创建一个空字典

>>> info= {'000001': '上证指数', '399001': '深证成指'}

>>> info['000300']= '沪深300' #新增元素

>>> info['000001']= '上证指数' #修改元素值

>>> del info['399001'] #删除元素值

# 课堂练习

正确定义一个字典的是 ( )

A、 `a=["A": 10, "B": 20, "C": 30]`

B、 `a=("A": 10, "B": 20, "C": 30)`

C、 `a={A:10, B: 20, C: 30}`

D、 `a={"A":10, "B":20, "C": 30}`

# 常用字典方法

**di.keys():** 返回包含字典所有**键**的列表

**di.values():** 返回包含字典所有**值**的列表

**di.items():** 返回包含所有(**键**、**值**)项列表

**di.get(key,[default]):** 返回键**key**对应的**值**，若  
key不存在，则返回default

**di.update(a):** 将字典a中的键值对添加到di中

# 课堂练习

若dic1 = {'甲':3, '乙':1, '丙':5, '丁':8}, 则执行  
print(dic1.get('乙', '未找到'))的结果是 ( )

A、未找到

B、1

C、报错

D、输出空值

# 词云相关库

**matplotlib:** 用于绘图的第三方库

**wordcloud:** 用于词云展示的第三方库

**imageio:** 读取和写入各种图像的第三方库

# 爱心词云

	A	B
1	学号	姓名
2	20002518	邹轩敏
3	21012973	李一凡
4	21012974	王绘雯
5	21012975	黄佳妮
6	21012976	李雨杉
7	21012978	胡凯
8	21012979	党嘉懿
9	21012980	马睿
10	21012981	赵骏飞
11	21012982	袁洲力



```

from random import *
counts={ }#创建一个空字典
with open("student.csv", 'r') as fobj:
    for i in fobj:
        if i[:2]=="学号":
            continue
        i=i. strip()
        code, name=i. split(",")
        counts[name]=randint(30,100)

import matplotlib. pyplot as plt
from wordcloud import WordCloud
from imageio.v2 import imread

```

# 爱心词云

```
pic = imread('love.png')
wc=WordCloud(mask=pic,font_path='msyh.ttc', #中文字体
              repeat=False, #内容是否可以重复
              background_color='white', #设置背景颜色
              max_words=100, #设置最大词数
              max_font_size=120, #设置字体最大值
              min_font_size=10, #设置字体最小值
              random_state=50, #设置有配色方案
              scale=1) #按照比例进行放大画布

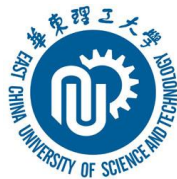
wc.generate_from_frequencies(counts)
plt.imshow(wc)
plt.show()
```



[illegible]

# 案例分析





谢 谢