1 数据分析

1.1 Python基础

1.2 标题设置用#号，需要空格

1.2.0.1 Python里面的一些基本操作

这里显示粗体文字用两颗星星，文本前后都得是两颗

其他的不是粗体

如果是斜体用一颗星

如果是斜体加粗是三颗星星

如果要删除内容，可以用文本前后两个波浪线

比如这~~一行是要删除的内容~~

一个大于号表示引用

> 这一行文字引用

可以用三个以上的减号+空行来表示分割线

比如

分割这一行文字

也可以用三个以上的星号表示分割线

比如

星号和文本之间不需要加空格

1.2.0.2 表格语法

表头	表头	表头
内容	内容	内容
内容	内容	内容
内容	内容	内容

表头	表头	表头
内容	内容	内容
内容	内容	内容
内容	内容	内容

姓名	语文	数学	英语	体育	音乐
张三	88	99	100	98	78
李四	94	87	92	88	99
王五	87	89	98	93	97
赵六	92	94	85	89	92

1.2.0.3 超链接的使用，方括号和小括号都是英文状态下的括号

百度一下你就知道，有事找度娘 #方括号里面为自己定义的名字，小括号里面是想要链接的地址

1.2.0.4 不需要用的代码块

不需要运行的代码块用英文状态下的三个点括起来，即前后都需要加三个点，是esc下面的那个点，而且是在Markdown的模式下

```
if 条件：
    满足条件执行的操作
else
    不满足条件执行的操作
```

1.3 Python代码语法

1.3.1 查看帮助文档

1、print() #最常用的方法就是在括号状态下按下shift+tap键

意思就是要查找print的用法和括号里的参数，可以打出print()光标放在括号里，此时是代码模式，然后按下shift+tap键，就会出现用法

```
print()
```

2、也可以用"？print"，此时也是在代码模式下

```
?print
```

3、还可以help(print)，此时也是代码模式

```
help(print)
```

```
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
```

```
for i in range(10):
    print()
```

1.3.2 Python的注释功能

用#增加注释 比如： print('a') #打印字母a

```
print('a') #打印字母a,代码模式下进行执行
```

```
a
```

多行注释如果使用三个引号的话，不会按照正常逻辑执行代码，所以不常用 ''' print(123) print(abc) print(2q3) '''

```
'''
print(123)
print(abc)
print(2q3)
'''
```

```
'\nprint(123)\nprint(abc)\nprint(2q3)\n'
```

上面'\n'的意思是换行了

```
所以多行注释可以将代码写完后，全选，然后按Ctrl+? ，进行批量注释，可以实现正常逻辑执行代码
```

1.3.3 全局输出控制语句

```
# 设置全部行输出
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

> 上述代码表示，在输出结果时，以单元格为单位，令所有可以被输出的对象，依次进行输出

```
x="我"
y="你"
z="她"

x
y #没有设置全局输出时，默认只输出需要输出结果的最后一个
```

'你'

```
x="我"
y="你"
z="她"

x
y
z#没有设置全局输出时，默认只输出需要输出结果的最后一个
```

```
x="我"
y="你"
z="她"

x
y#设置全局输出后，输出需要输出结果的全部
```

'我'

'你'

```
x="我"
y="你"
z="她"

x
y
z#设置全局输出后，输出需要输出结果的全部
```

'我'

'你'

'她'

> 如果设置了全局输出后，不想全部输出了，只能重启内核，所以执行全局输出代码的时候要考虑清楚是否后面的代码都是全部输出的

1.3.4 Python的print() 功能

```
print('小明')
```

小明

```
print('a')
```

a

```
print(a)  #a没有引号代表变量，没有给变量进行定义，所以输出会报错
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_26588/3191407592.py in <module>
----> 1 print(a)  #a没有引号代表变量，没有给变量进行定义，所以输出会报错

NameError: name 'a' is not defined
```

```
print()
```

```
>Docstring:
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

sep里的参数是控制一个print内的多个值以什么进行连接,它的默认值是空格,当不设置的时候,一个print内的多个值以空格进行连接

end参数是控制多个print之间输出时以什么进行连接,默认是'\n'就是换行
```

```
print('大明','小明')
```

大明 小明

```
print('大明','小明')
print('大明','小明',sep='**')
print('大明','小明')#sep的练习
```

```
大明 小明
大明**小明
大明 小明
```

```
print('大明','小明',end='加上')
print('大明','小明',sep='**')
print('大明','小明',sep='和')#sep和end的练习
```

```
大明 小明加上大明**小明
大明和小明
```

1.3.5 input()获取用户输入

> input()语句可以获取一个用户输入的字符串

```
input()
```

```
1
```

'1'

> input()进行执行后会出现一个黑色小长条,必须在这个长条里面输入一个值,这个值会以字符串的格式输出

如果不设置一个值,input()前面会显示*号,代表正在运行,但是一直运行不完,结束不了,后面所有的代码都无法运行

如果出现了这种情况,就查询前面的代码看是否有input()语句没有赋值,填上一个值看是否可以继续运行, 如果上述步骤仍然显示*号不能正常进行运行,就是卡住了,可以按照下面步骤进行操作 1.可以按左上角功能区的黑色停止的钮,2.第一步还不行就按重启的钮,3.第二步还不行就只能关闭jupyter notebook,进行重新打开

```
a=input()
```

你好

```
a
```

'你好'

```
b=input('请输入您的密码')
```

请输入您的密码1234567

```
b
```

'1234567'

1.3.6 Python中的变量

```
a=1 #定义了一个叫做a的变量,指向了1,也就是说存储了一个a=1的记录,后面调用a这个变量就会输出对应的1这个值
```

```
a
```

1

```
print(a)
```

1

```
print('a') #用引号引起来的表示是字符串,不是变量a
```

a

1.3.6.1 单个变量的赋值

```
b=2
```

```
print(b)
```

2

```
d#这个d的变量没有被赋值,所以执行的时候就会被报错
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_26588/3293192696.py in <module>
----> 1 d#这个d的变量没有被赋值,所以执行的时候就会被报错

NameError: name 'd' is not defined
```

1.3.6.2 多变量同时赋值

```
x=y=z=2#链式赋值，赋值之后这几个变量对应的值是相同的，并且它们的id也是相同的
```

```
x
y
z
```

```
2
```

```
2
```

```
2
```

```
id(x)
id(y)
id(z)
```

```
2116197968208
```

```
2116197968208
```

```
2116197968208
```

1.3.6.3 拆包式赋值

```
m,n,j=5,8,'abc'#拆包式赋值，变量之间用逗号隔开，值之间用逗号隔开，字符串用引号引起来，它们会自动的按照位置一一对应
```

```
m
n
j
```

```
5
```

```
8
```

```
'abc'
```

```
m
j
```

```
5
```

```
'abc'
```

```
n
j
```

```
8
```

```
'abc'
```

1.3.6.4 交换两个变量的赋值

```
m, n=n, m
```

```
m
n
```

```
8
```

```
5
```

1.3.6.5 变量可以被重复赋值

```
a
```

```
1
```

```
a=5
a
```

```
5
```

```
a=' 你好'
a
```

```
' 你好'
```

1.3.6.6 变量名区分大小写

```
A#A并没有被定义为一个变量,所以会报错,大小写的a可以代表不同的变量
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_26588/378453753.py in <module>
----> 1 A

NameError: name 'A' is not defined
```

```
A=' 我爱你'
```

```
A
```

```
' 我爱你'
```

```
a
A
```

```
' 你好'
```

```
' 我爱你'
```

1.3.6.7 变量名的命名规范

> 变量名只能包含字母,数字,下划线 变量名可以以字母和下划线开头,但是不能以数字开头,尽量不要用下划线做开头 Python中的关键字和函数名不可以作为变量名 实际工作中,变量的命名尽量能体现变量的性质,比如score_class5 这个作为一个变量名说明变量被赋值的是5班的成绩

```
a3='我爱中国'
```

```
a3
```

```
'我爱中国'
```

```
3a=5 #报错,因为数字不能作为变量名的开头
```

```
  File "C:\Users\99253\AppData\Local\Temp/ipykernel_26588/2672673897.py", line 1
    3a=5 #报错,因为数字不能作为变量名的开头
     ^
SyntaxError: invalid syntax
```

```
if=5#也会报错,因为if是Python里面的关键字
```

```
  File "C:\Users\99253\AppData\Local\Temp/ipykernel_26588/2464191989.py", line 1
    if=5#也会报错,因为if是Python里面的关键字
       ^
SyntaxError: invalid syntax
```

```
import keyword
keyword.kwlist  #查看Python中的关键字
```

```
['False',
 'None',
 'True',
 '__peg_parser__',
 'and',
```

```
while
if
elif   #绿色高亮的词汇就是关键字
anc
bd
cm    #定义的变量是黑色显示的
list
print  #Python中的函数也是绿色显示,但不一定高亮
```

```
  File "C:\Users\99253\AppData\Local\Temp/ipykernel_26588/592224721.py", line 1
    while
         ^
SyntaxError: invalid syntax
```

print是一个函数名,但是被作为变量名进行定义了一个变量之后,他的函数功能就丧失了,

如果想恢复函数功能,用del print 可以删除叫做print变量的功能,从而恢复print的函数的功能

```
print=1#这是print被定义成了一个变量
```

```
print
```

```
1
```

```
del print #这个操作是删除print的变量名的功能,恢复print函数的功能
```

```
print(123)
```

```
123
```

1.3.7 Python语句的书写规范

Python是没有结束符的,不像SQL用分号结束一个语句进行分割

1.3.7.1 Python语句的编写规则

第一种:一句一行

```
a=1
```

```
a
```

```
1
```

```
a=1
b=2

a
b
#不想挨着就可以打空行,不会影响最终的执行命令
```

```
1
```

```
2
```

第二种:一行多句,需要加分号隔开

```
a=1;b=2;c=3
a;b;c    #这种写法不会报错,但也不常用
```

```
1
```

```
2
```

```
3
```

```
a=1;b=2;c=3
a;b
c    #这种写法不会报错,但也不常用
```

```
1
```

```
2
```

```
3
```

第三种:一句多行,需要在每一行的结尾用续行符进行续行,续行符是反斜杠
或者也可以用三个引号,也可以用 \n 在实际工作中用三个引号的时候居多

```
print('我有一个梦想,希望我能快乐开心的活着,希望这个世界依然美好,希望人心依然善良,希望所有的人都能梦想成真,\
如果时间可以倒流,定不辜负年少青春,回首过往的点滴,开心,痛苦,疾苦,隐忍,都是人生最好的体验,时过境迁,现实社会太过残酷,\
现实社会,生活不易,还需保持良好心态,坚持初心,勇敢前进,不管世界多么混杂,做好自己内心纯净的善良,问心无愧就好')
#续行符是这几行接在一起形成完整的一个段落,不会因为用了续行符而自动换行
```

```
我有一个梦想,希望我能快乐开心的活着,希望这个世界依然美好,希望人心依然善良,希望所有的人都能梦想成真,如果时间可以倒流,定不辜负年少青春,回首过往的点滴,开心,痛苦,疾苦,隐忍,都是人生最好的体验,时过境迁,现实社会太过残酷,现实社会,生活不易,还需保持良好心态,坚持初心,勇敢前进,不管世界多么混杂,做好自己内心纯净的善良,问心无愧就好
```

```
print('''我有一个梦想,希望我能快乐开心的活着,希望这个世界依然美好,希望人心依然善良,
希望所有的人都能梦想成真,
如果时间可以倒流,定不辜负年少青春,回首过往的点滴,开心,痛苦,疾苦,隐忍,都是人生最好的体验,
时过境迁,现实社会太过残酷,
现实社会,生活不易,还需保持良好心态,坚持初心,勇敢前进,
不管世界多么混杂,做好自己内心纯净的善良,问心无愧就好''')
#三引号是会根据函数里面的内容的换行格式进行换行,与函数里面的内容保持一致的格式.与 \n 的功能一样
```

```
我有一个梦想,希望我能快乐开心的活着,希望这个世界依然美好,希望人心依然善良,
希望所有的人都能梦想成真,
如果时间可以倒流,定不辜负年少青春,回首过往的点滴,开心,痛苦,疾苦,隐忍,都是人生最好的体验,
时过境迁,现实社会太过残酷,
现实社会,生活不易,还需保持良好心态,坚持初心,勇敢前进,
不管世界多么混杂,做好自己内心纯净的善良,问心无愧就好
```

```
print('我有一个梦想,希望我能快乐开心的活着,希望这个世界依然美好,希望人心依然善良,\n希望所有的人都能梦想成真,\n如果时间可以倒流,定不
#这个是用 \n 进行的换行操作
```

```
我有一个梦想,希望我能快乐开心的活着,希望这个世界依然美好,希望人心依然善良,
希望所有的人都能梦想成真,
如果时间可以倒流,定不辜负年少青春,回首过往的点滴,开心,痛苦,疾苦,隐忍,都是人生最好的体验,
时过境迁,现实社会太过残酷,
现实社会,生活不易,还需保持良好心态,坚持初心,勇敢前进,
不管世界多么混杂,做好自己内心纯净的善良,问心无愧就好
```

1.3.8 数据类型

1.3.8.1 数据类型分类

数字(numbers):整数(null),浮点数(float)(小数点),复数(complex)(不常用)

布尔(booleans):TRUE和FALSE

字符串(strings):uniconde字符序列,用引号引起来

列表(list):有序的值的序列,用方括号[]

元组(tuples):有序的值的序列且不可改变,圆括号()或者逗号,

字典(dictionaries):无序的键值对的组合,花括号{}和key

集合(sets):无序的不含重复值的序列

1.3.8.2 查看数据类型的函数---type

```
a=1
type(a)
```

```
int
```

```
type([1,2,3,4])
```

```
list
```

```
type(3.5)
```

```
float
```

```
type({2:'a'})
```

```
dict
```

1.3.8.3 转换数据类型

> 使用强制类型转换函数

```
a
```

```
1
```

```
float(a)
```

```
1.0
```

```
a=float(a) #如果想使a这个变量引用转换后的数据类型,可以再重新定义a这个变量等于float(a)
```

```
a
```

```
1.0
```

```
b='2.5'
type(b)  #这里的2.5是被定义为字符串的,因为用引号引起来了
```

```
str
```

```
int(b)  #报错的原因是2.5本身是浮点数(小数),没有办法强制转换成整数,所以转换的数据类型要与本身的特征一致
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_26588/3956108814.py in <module>
----> 1 int(b)

ValueError: invalid literal for int() with base 10: '2.5'
```

```
float(b)
```

```
2.5
```

```
c='5'
int(c)  #5本身是整数,所以可以从字符串型转换成整数型
```

```
5
```

```
int('你好')  #报错是因为'你好'本身是字符串,不具有数字,所以无法转换成数值型
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_26588/3708755376.py in <module>
----> 1 int('你好')

ValueError: invalid literal for int() with base 10: '你好'
```