章节一：python 使用 openpyxl 操作 excel

1、python 怎么打开及读取表格内容？

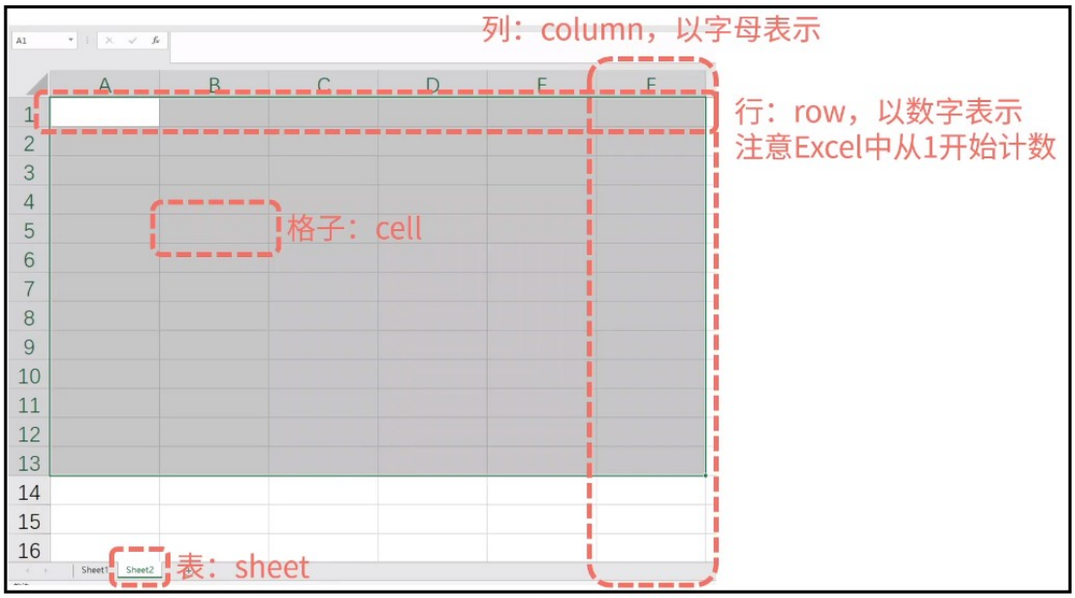
* openpyxl 最好用的 python 操作 excel 表格库，不接受反驳；

* openpyxl 官网链接：https://openpyxl.readthedocs.io/en/stable/；

* openpyxl 只支持【.xlsx / .xlsm / .xltx / .xltm】格式的文件；

1）Excel 表格述语

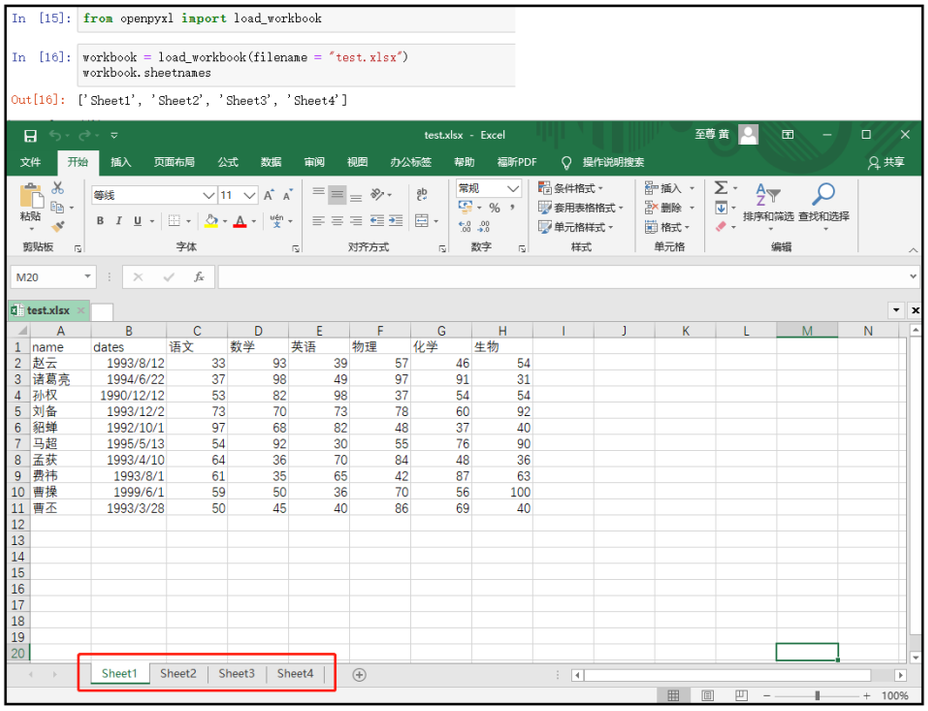
这里需要大家仔细查看图中的每一项内容，知道什么是“行(row)、列(column)”？什么是“格子(cell)”？什么是“sheet 表”？



2）打开 Excel 表格并获取表格名称

```
from openpyxl import load_workbook
workbook = load_workbook(filename = "test.xlsx")
workbook.sheetnames
```

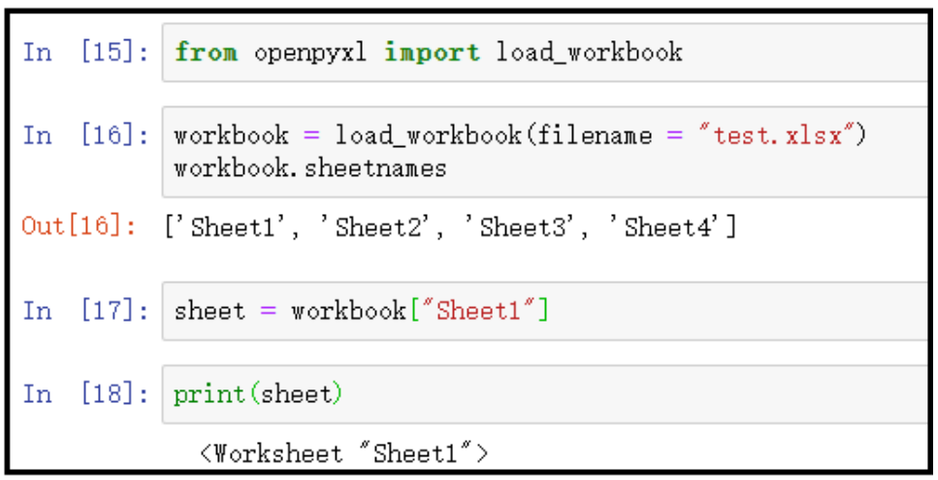
结果如下：



3）通过 sheet 名称获取表格

```
from openpyxl import load_workbook

workbook = load_workbook(filename = "test.xlsx")

workbook.sheetnames

sheet = workbook["Sheet1"] print(sheet)
```

结果如下：



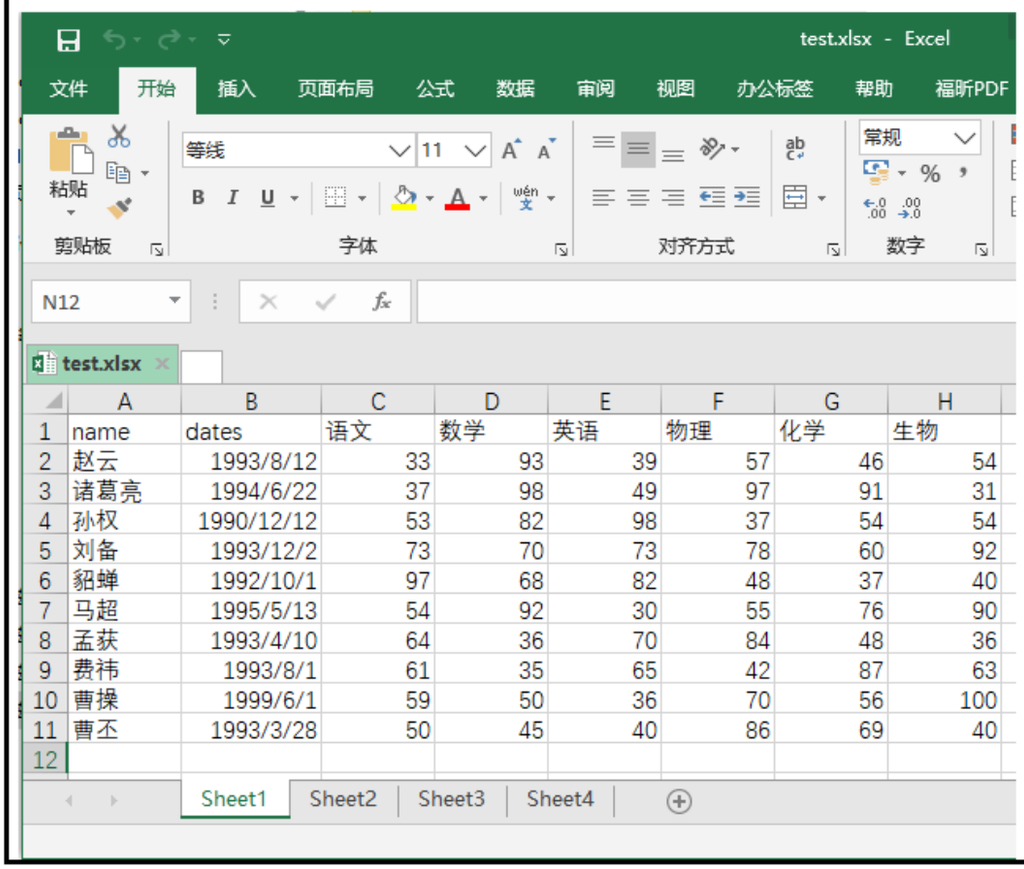
4）获取表格的尺寸大小(几行几列数据)

这里所说的尺寸大小，指的是 excel 表格中的数据有几行几列，针对的是不同的 sheet 而言。

```
sheet.dimensions
```

结果如下：

```
In [15]: from openpyxl import load_workbook

In [16]: workbook = load_workbook(filename = "test.xlsx")
         workbook.sheetnames

Out[16]: ['Sheet1', 'Sheet2', 'Sheet3', 'Sheet4']

In [17]: sheet = workbook["Sheet1"]

In [18]: print(sheet)

         <Worksheet "Sheet1">

In [19]: sheet.dimensions

Out[19]: 'A1:H11'
```



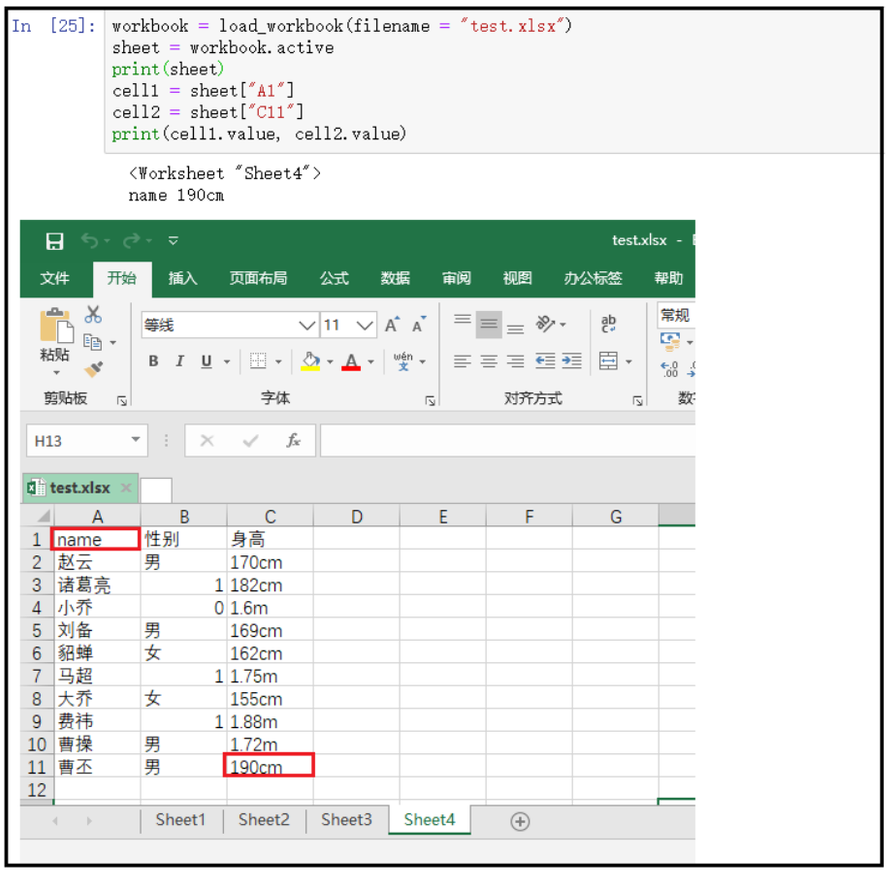
	A	B	C	D	E	F	G	H
1	name	dates	语文	数学	英语	物理	化学	生物
2	赵云	1993/8/12	33	93	39	57	46	54
3	诸葛亮	1994/6/22	37	98	49	97	91	31
4	孙权	1990/12/12	53	82	98	37	54	54
5	刘备	1993/12/2	73	70	73	78	60	92
6	貂蝉	1992/10/1	97	68	82	48	37	40
7	马超	1995/5/13	54	92	30	55	76	90
8	孟获	1993/4/10	64	36	70	84	48	36
9	费祎	1993/8/1	61	35	65	42	87	63
10	曹操	1999/6/1	59	50	36	70	56	100
11	曹丕	1993/3/28	50	45	40	86	69	40
12								

5）获取表格内某个格子的数据

① sheet["A1"]方式

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active print(sheet)
cell1 = sheet["A1"]
cell2 = sheet["C11"]
print(cell1.value, cell2.value)
"""
workbook.active  打开激活的表格；
sheet["A1"]  获取 A1 格子的数据；
cell.value  获取格子中的值；
"""
```

结果如下：



② sheet.cell(row=, column=)方式

*下面这种方式更简单，大家可以对比这两种方式；

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
print(sheet)
cell1 = sheet.cell(row = 1,column = 1)
cell2 = sheet.cell(row = 11,column = 3)
print(cell1.value, cell2.value)
```

结果如下：

```
In [32]: workbook = load_workbook(filename = "test.xlsx")
         sheet = workbook.active
         print(sheet)
         cell1 = sheet["A1"]
         cell2 = sheet["C11"]
         print(cell1.value, cell2.value)

         <Worksheet "Sheet4">
         name 190cm

In [33]: workbook = load_workbook(filename = "test.xlsx")
         sheet = workbook.active
         print(sheet)
         cell1 = sheet.cell(row = 1, column = 1)
         cell2 = sheet.cell(row = 11, column = 3)
         print(cell1.value, cell2.value)

         <Worksheet "Sheet4">
         name 190cm
```

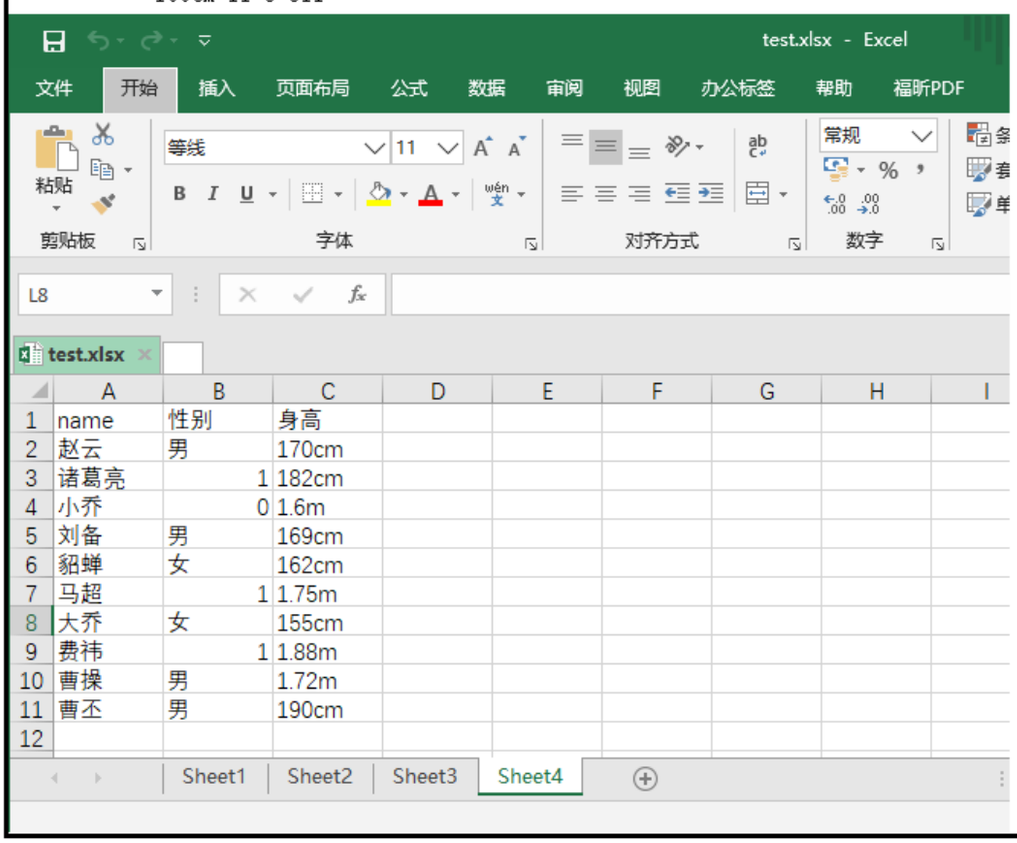
6）获取某个格子的行数、列数、坐标

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
print(sheet)
cell1 = sheet["A1"]
cell2 = sheet["C11"]
```

```
print(cell1.value, cell1.row, cell1.column, cell1.coordinate)
print(cell2.value, cell2.row, cell2.column, cell2.coordinate)
"""
.row  获取某个格子的行数；
.columns  获取某个格子的列数；
.corordinate  获取某个格子的坐标；
 """
```

结果如下：

```
In [27]: workbook = load_workbook(filename = "test.xlsx")
         sheet = workbook.active
         print(sheet)
         cell1 = sheet["A1"]
         cell2 = sheet["C11"]
         print(cell1.value, cell1.row, cell1.column, cell1.coordinate)
         print(cell2.value, cell2.row, cell2.column, cell2.coordinate)

         <Worksheet "Sheet4">
         name 1 A A1
         190cm 11 C C11
```



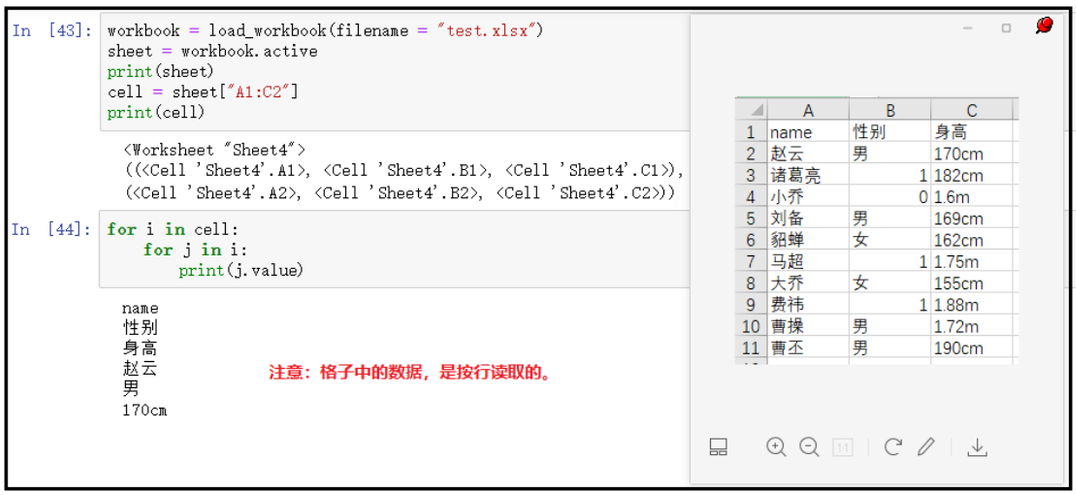
	A	B	C
1	name	性别	身高
2	赵云	男	170cm
3	诸葛亮	1	182cm
4	小乔	0	1.6m
5	刘备	男	169cm
6	貂蝉	女	162cm
7	马超	1	1.75m
8	大乔	女	155cm
9	费祎	1	1.88m
10	曹操	男	1.72m
11	曹丕	男	190cm
12			

7）获取一系列格子

① sheet[]方式

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active print(sheet)
# 获取 A1:C2 区域的值
cell = sheet["A1:C2"]
print(cell)
for i in cell:
    for j in i:
        print(j.value)
```

结果如下：



特别的：如果我们只想获取“A 列”，或者获取“A-C 列”，可以采取如下方式：

```
sheet["A"] --- 获取 A 列的数据
sheet["A:C"] --- 获取 A,B,C 三列的数据
sheet[5] --- 只获取第 5 行的数据
```

② .iter_rows()方式

* 当然有.iter_rows()方式，肯定也会有.iter_cols()方式，只不过一个是按行读取，一个是按列读取。

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active print(sheet)
# 按行获取值
for i in sheet.iter_rows(min_row=2, max_row=5, min_col=1, max_col=2):
    for j in i:
        print(j.value)
# 按列获取值
for i in sheet.iter_cols(min_row=2, max_row=5, min_col=1, max_col=2):
    for j in i:
        print(j.value)
```

结果如下：

```
In [63]: workbook = load_workbook(filename = "test.xlsx")
         sheet = workbook.active
         print(sheet)

         for i in sheet.iter_rows(min_row=2, max_row=5, min_col=1, max_col=2):
             for j in i:
                 print(j.value)

<Worksheet "Sheet4">
赵云
男
诸葛亮          按行读取数据
1
小乔
0
刘备
男

In [64]: for i in sheet.iter_cols(min_row=2, max_row=5, min_col=1, max_col=2):
             for j in i:
                 print(j.value)

赵云
诸葛亮
小乔            按列读取数据
刘备
男
1
0
男
```

	A	B	C
1	name	性别	身高
2	赵云	男	170cm
3	诸葛亮	1	182cm
4	小乔	0	1.6m
5	刘备	男	169cm
6	貂蝉	女	162cm
7	马超	1	1.75m
8	大乔	女	155cm
9	费祎	1	1.88m
10	曹操	男	1.72m
11	曹丕	男	190cm

③ sheet.rows()

* 帮助我们获取所有行

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
print(sheet)
for i in sheet.rows:
    print(i)
```

结果如下：

```
In [68]: workbook = load_workbook(filename = "test.xlsx")
         sheet = workbook.active
         print(sheet)

         for i in sheet.rows:
             print(i)

<Worksheet "Sheet4">                    一共11行、3列数据
(<Cell 'Sheet4'.A1>, <Cell 'Sheet4'.B1>, <Cell 'Sheet4'.C1>)
(<Cell 'Sheet4'.A2>, <Cell 'Sheet4'.B2>, <Cell 'Sheet4'.C2>)
(<Cell 'Sheet4'.A3>, <Cell 'Sheet4'.B3>, <Cell 'Sheet4'.C3>)
(<Cell 'Sheet4'.A4>, <Cell 'Sheet4'.B4>, <Cell 'Sheet4'.C4>)
(<Cell 'Sheet4'.A5>, <Cell 'Sheet4'.B5>, <Cell 'Sheet4'.C5>)
(<Cell 'Sheet4'.A6>, <Cell 'Sheet4'.B6>, <Cell 'Sheet4'.C6>)
(<Cell 'Sheet4'.A7>, <Cell 'Sheet4'.B7>, <Cell 'Sheet4'.C7>)
(<Cell 'Sheet4'.A8>, <Cell 'Sheet4'.B8>, <Cell 'Sheet4'.C8>)
(<Cell 'Sheet4'.A9>, <Cell 'Sheet4'.B9>, <Cell 'Sheet4'.C9>)
(<Cell 'Sheet4'.A10>, <Cell 'Sheet4'.B10>, <Cell 'Sheet4'.C10>)
(<Cell 'Sheet4'.A11>, <Cell 'Sheet4'.B11>, <Cell 'Sheet4'.C11>)
```

2、python 如何向 excel 中写入某些内容？

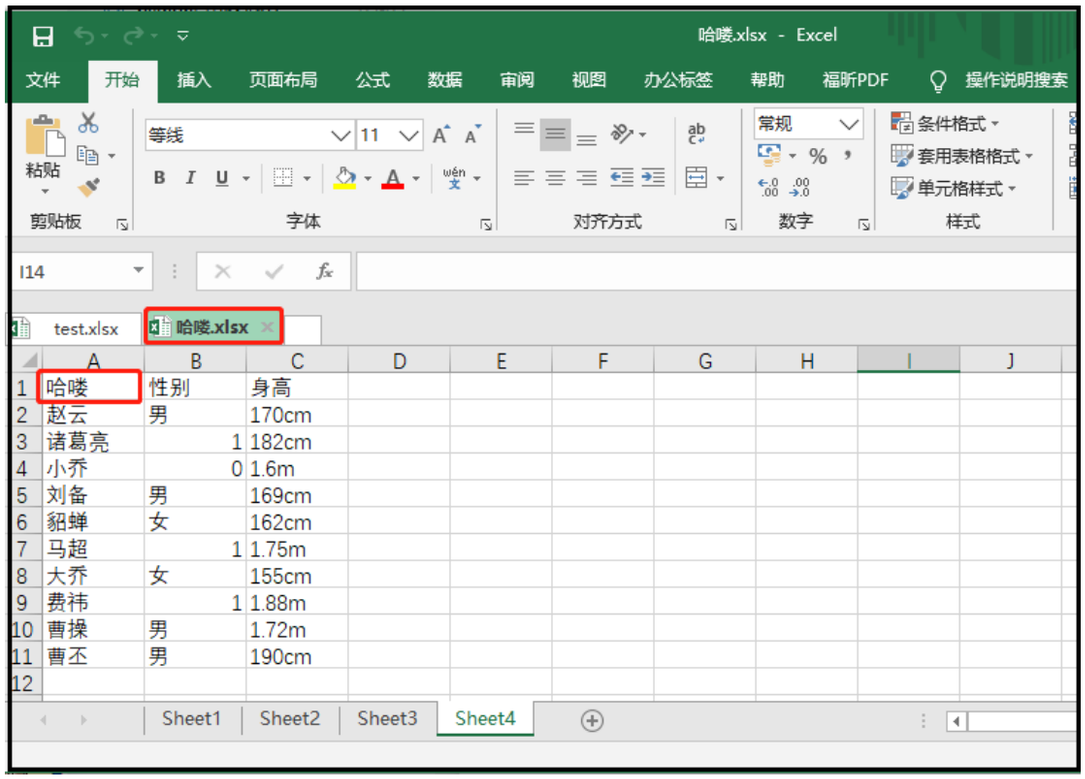
1）修改表格中的内容

① 向某个格子中写入内容并保存

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
print(sheet)
```

```
sheet["A1"] = "哈喽"
# 这句代码也可以改为 cell = sheet["A1"] cell.value = "哈喽"
workbook.save(filename = "哈喽.xlsx")
"""
注意：我们将“A1”单元格的数据改为了“哈喽”，并另存为了“哈喽.xlsx”文件。 如果我们保存的时候，不修改表名，相当于直接修改源文件；
"""
```

结果如下：



② .append()：向表格中插入行数据(很有用)

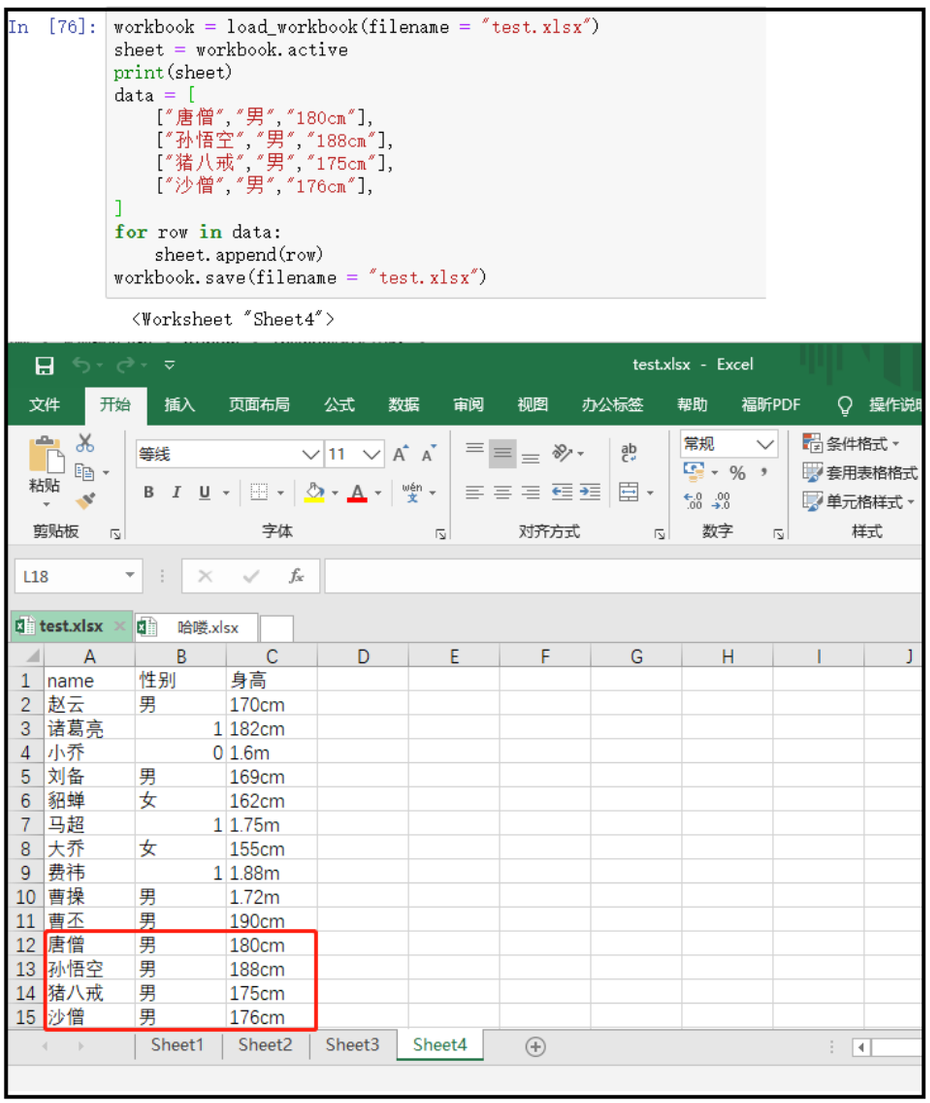
* .append()方式：会在表格已有的数据后面，增添这些数(按行插入)；

* 这个操作很有用，爬虫得到的数据，可以使用该方式保存成 Excel 文件；

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
```

```
print(sheet)
data = [
        ["唐僧","男","180cm"],
        ["孙悟空","男","188cm"],
        ["猪八戒","男","175cm"],
        ["沙僧","男","176cm"],
      ]
for row in data:
    sheet.append(row)
workbook.save(filename = "test.xlsx")
```

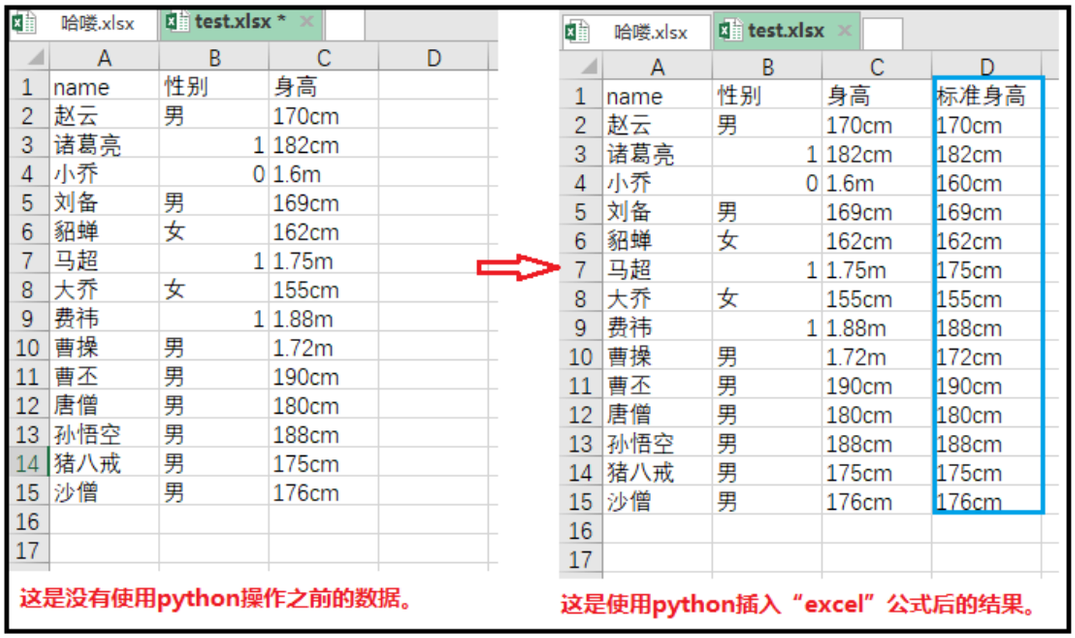
结果如下：



③ 在 python 中使用 excel 函数公式(很有用)

```
# 这是我们在 excel 中输入的公式
=IF(RIGHT(C2,2)="cm",C2,SUBSTITUTE(C2,"m",""))*100&"cm")
# 那么，在 python 中怎么插入 excel 公式呢？
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
print(sheet)
sheet["D1"] = "标准身高"
for i in range(2,16):
    sheet["D{}".format(i)] =
'=IF(RIGHT(C{},2)="cm",C{},SUBSTITUTE(C{},"m","")*100&"cm")'.format(i,i,i)
workbook.save(filename = "test.xlsx")
```

结果如下：



此时，你肯定会好奇，python 究竟支持写哪些“excel 函数公式”呢？我们可以使用如下操作查看一下。

```
import openpyxl
from openpyxl.utils import FORMULAE
print(FORMULAE)
```

结果如下：

```
In [84]: import openpyxl
         from openpyxl.utils import FORMULAE
         print(FORMULAE)
```

截取部分如下

```
frozenset({'MIN', 'FV', 'BIN2OCT', 'PEARSON', 'OCT2DEC', 'COUPDAYBS', 'EVEN', 'ACOSH', 'IMSUB', 'DGET', 'AMORLINC', 'VARP', 'INTERCEP
T', 'ASIN', 'RANK', 'OFFSET', 'RATE', 'ISBLANK', 'DSUM', 'QUOTIENT', 'COUPNCD', 'NORMINV', 'LEN', 'YIELD', 'CUBESETCOUNT', 'HLOOKUP',
'DURATION', 'MIRR', 'LOGNORMDIST', 'TRUNC', 'COUNTIFS', 'ODDLPRICE', 'AND', 'ZTEST', 'XIRR', 'IMSIN', 'ATAN2', 'ATANH', 'TIME', 'EOMONT
H', 'DEGREES', 'IMEXP', 'CHIDIST', 'COMPLEX', 'AVEDEV', 'RADIANS', 'TEXT', 'STDEVPA STEYX', 'TBILLYIELD', 'MROUND', 'DELTA', 'GAMMADIS
T', 'PRICEMAT', 'SUM', 'CRITBINOM', 'COUPPCD', 'EFFECT', 'WORKDAY.INTL', 'REPLACEB', 'FTEST', 'DEC2BIN', 'FALSE', 'GESTEP', 'MULTINOMIA
L', 'T', 'PHONETIC', 'ISERR', 'ASC', 'HOUR', 'RECEIVED', 'HYPERLINK', 'CEILING', 'ISPMT', 'RANDBETWEEN', 'CONVERT', 'CUBERANKEDMEMBER',
'INDIRECT', 'PPMT', 'WORKDAY ', 'DEVSQ', 'AMORDEGRC', 'DATE', 'INT', 'ERF', 'ERROR.TYPE', 'NA', 'SUMXMY2', 'GETPIVOTDATA', 'DOLLARDE',
'BESSELI', 'IMCONJUGATE', 'FACTDOUBLE', 'VAR', 'TREND', 'PERCENTRANK', 'MINUTE', 'IMABS', 'INTRATE', 'IMCOS', 'TRUE ADDRESS', 'CUBESE
T', 'IMREAL', 'ROMAN', 'ACOS', 'NETWORKDAYS.INTL', 'LCM', 'SQRT', 'AVERAGEA', 'SUBTOTAL', 'RAND', 'LOGINV', 'ODDFPRICE', 'MODE', 'BESSE
LK', 'N', 'HEX2OCT', 'TINV', 'RIGHT', 'OR', 'GCD', 'EDATE', 'SYD', 'ERFC', 'IMLOG2', 'IMPOWER', 'AVERAGE', 'DVARP', 'OCT2HEX', 'DSTDE
V', 'DAYS360', 'ODDFYIELD', 'NOW', 'MONTH', 'PV', 'CLEAN', 'BINOMDIST', 'AVERAGEIF', 'COLUMN', 'EXACT', 'YEARFRAC', 'IMLN', 'SUMX2PY2',
'KURT', 'TDIST', 'IMSUM', 'ODDLYIELD', 'INFO', 'ISNA', 'NORMDIST', 'EXP', 'TANH', 'GAMMALN', 'MAX', 'DISC', 'SUMIFS', 'SERIESSUM', 'SUM
X2MY2', 'LENB', 'CUBEMEMBER', 'DB', 'DMAX', 'BIN2DEC', 'TYPE', 'ASINH', 'FVSCHEDULE', 'DATEDIF', 'IMDIV', 'LARGE', 'DEC2OCT', 'FIND',
'COUNTBLANK', 'SUMSQ', 'LOOKUP', 'ACCRINT', 'HYPGEOMDIST', 'AREAS', 'CUBEKPIMEMBER', 'MINA', 'DSTDEVP', 'EXPONDIST', 'FACT', 'YIELDDIS
C', 'MOD', 'CONFIDENCE', 'ISEVEN', 'LEFT', 'SECOND', 'ABS', 'PRODUCT', 'COMBIN', 'DVAR', 'MMULT', 'PROB', 'TRIMMEAN', 'CUBEVALUE', 'SKE
W', 'ISREF', 'BESSELY', 'MIDB', 'BETAINV', 'JIS', 'ISNONTEXT', 'SUBSTITUTE', 'COVAR', 'CONCATENATE', 'ROUNDUP', 'ROUND', 'DMIN', 'YIELD
MAT', 'ACCRINTM', 'TODAY', 'MINVERSE', 'AVERAGEIFS', 'PI', 'FIXED', 'TIMEVALUE', 'IF', 'IMLOG10', 'IFERROR', 'DEC2HEX', 'GROWTH', 'SIG
N', 'NPER', 'MAXA', 'PRICEDISC', 'COUPNUM', 'NORMSINV', 'VARA', 'DPRODUCT', 'SIN', 'CHAR', 'VDB', 'LEFTB', 'LOG', 'INDEX', 'PRICE', 'NO
T', 'TRANSPOSE', 'PERMUT', 'ODD', 'VLOOKUP', 'REPLACE', 'FDIST', 'COUPDAYS', 'MDURATION', 'FISHER', 'IMPRODUCT', 'DOLLAR', 'PROPER', 'S
UMIF', 'HEX2BIN', 'WEEKNUM', 'TAN', 'IMARGUMENT', 'FINV', 'VALUE', 'BETADIST', 'ROUNDDOWN', 'CHITEST', 'NEGBINOMDIST', 'ISERROR', 'STDE
V STDEVA', 'IMAGINARY', 'LOGEST', 'CUMPRINC', 'QUARTILE', 'COUPDAYSNC', 'NPV', 'MEDIAN', 'RIGHTB', 'DOLLARFR', 'COSH', 'TTEST', 'OCT2BI
N', 'COUNTA', 'YEAR', 'SLOPE', 'GEOMEAN', 'MATCH', 'TRIM', 'PMT', 'COUNTIF', 'LOWER', 'MID', 'VARPA', 'CELL', 'FREQUENCY', 'SMALL', 'DC
OUNT', 'SEARCHB', 'ISNUMBER', 'BAHTTEXT', 'POWER', 'REPT', 'MDETERM', 'HARMEAN', 'TBILLEQ', 'ROWS', 'ROW', 'CORREL', 'LOG10', 'ISO.CEIL
ING', 'SQRTPI', 'IMSQRT', 'SLN', 'PERCENTILE', 'CUMIPMT', 'ISODD', 'STDEVP', 'COS', 'FINDB', 'RSQ', 'SUMPRODUCT', 'TBILLPRICE', 'RTD',
'NORMSDIST', 'CODE', 'XNPV', 'DCOUNTA', 'FISHERINV', 'ISLOGICAL', 'UPPER', 'ISTEXT', 'COLUMNS', 'NETWORKDAYS', 'FLOOR', 'LINEST', 'GAMM
```

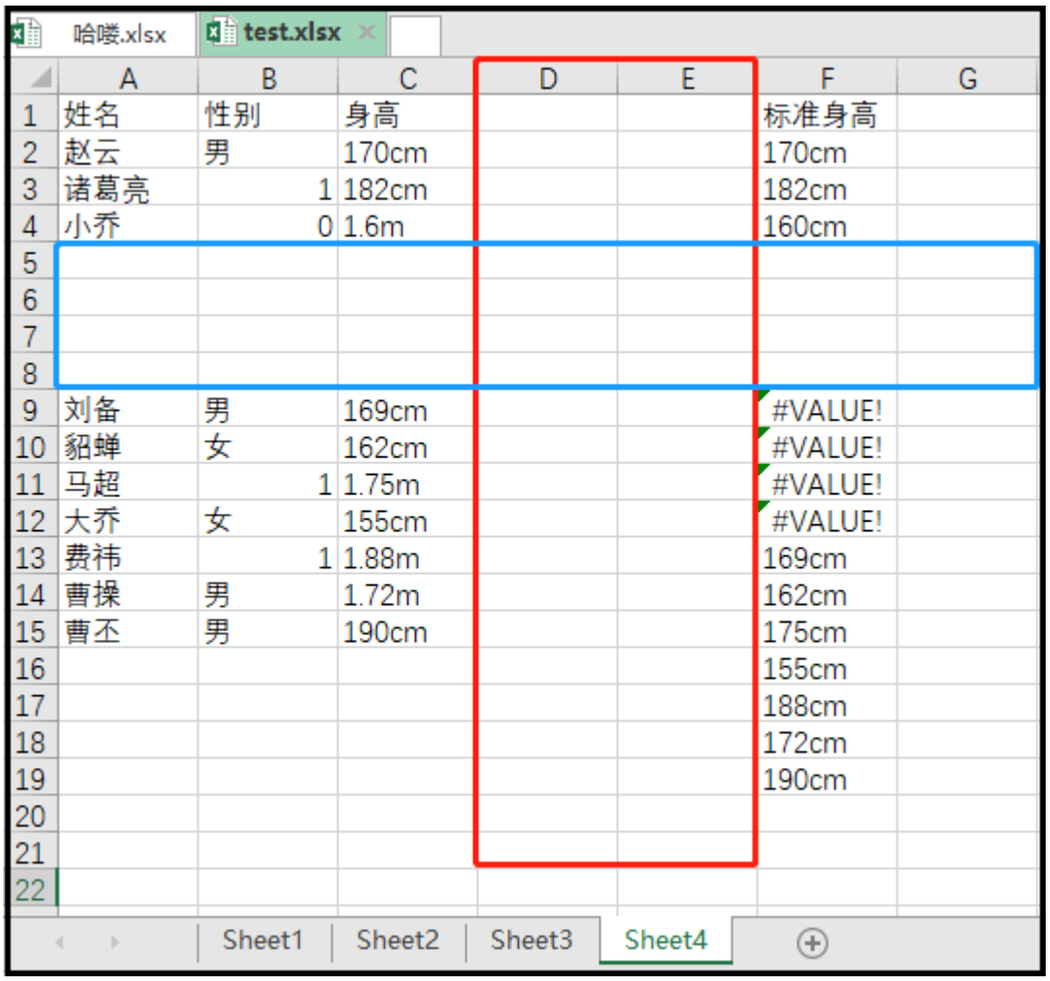
④ .insert_cols()和.insert_rows()：插入空行和空列

* .insert_cols(idx=数字编号, amount=要插入的列数)，插入的位置是在 idx 列数的左侧插入；

* .insert_rows(idx=数字编号, amount=要插入的行数)，插入的行数是在 idx 行数的下方插入；

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
print(sheet)
sheet.insert_cols(idx=4,amount=2)
sheet.insert_rows(idx=5,amount=4)
workbook.save(filename = "test.xlsx")
```

结果如下：



	A	B	C	D	E	F	G
1	姓名	性别	身高			标准身高	
2	赵云	男	170cm			170cm	
3	诸葛亮	1	182cm			182cm	
4	小乔	0	1.6m			160cm	
5							
6							
7							
8							
9	刘备	男	169cm			#VALUE!	
10	貂蝉	女	162cm			#VALUE!	
11	马超	1	1.75m			#VALUE!	
12	大乔	女	155cm			#VALUE!	
13	费祎	1	1.88m			169cm	
14	曹操	男	1.72m			162cm	
15	曹丕	男	190cm			175cm	
16						155cm	
17						188cm	
18						172cm	
19						190cm	
20							
21							
22							

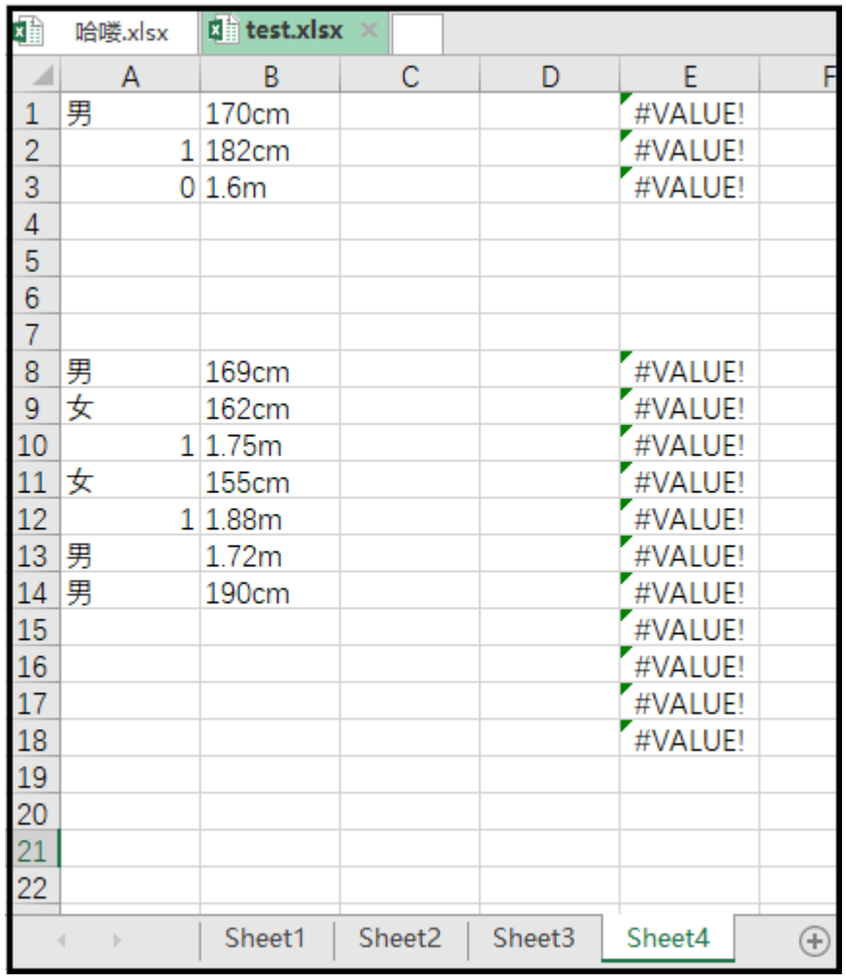
⑤ .delete_rows()和.delete_cols()：删除行和列

* .delete_rows(idx=数字编号, amount=要删除的行数)

* .delete_cols(idx=数字编号, amount=要删除的列数)

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active print(sheet)
# 删除第一列，第一行
sheet.delete_cols(idx=1)
sheet.delete_rows(idx=1)
workbook.save(filename = "test.xlsx")
```

结果如下：



	A	B	C	D	E
1	男	170cm			#VALUE!
2	1	182cm			#VALUE!
3	0	1.6m			#VALUE!
4					
5					
6					
7					
8	男	169cm			#VALUE!
9	女	162cm			#VALUE!
10	1	1.75m			#VALUE!
11	女	155cm			#VALUE!
12	1	1.88m			#VALUE!
13	男	1.72m			#VALUE!
14	男	190cm			#VALUE!
15					#VALUE!
16					#VALUE!
17					#VALUE!
18					#VALUE!
19					
20					
21					
22					

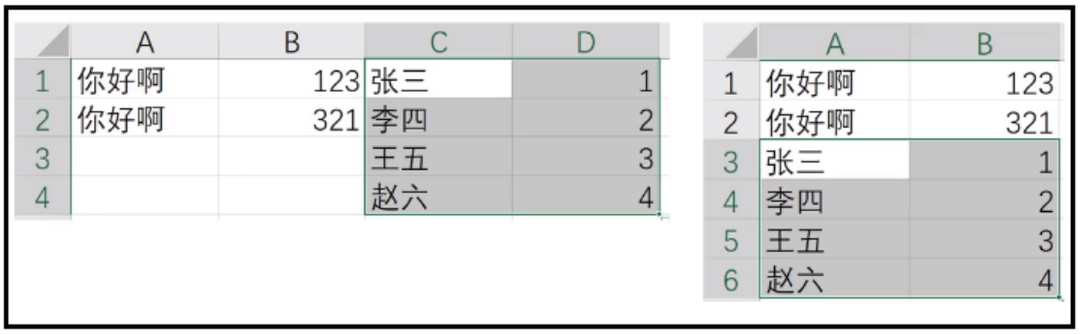
哈喽.xlsx　test.xlsx　Sheet1　Sheet2　Sheet3　Sheet4

⑥ .move_range()：移动格子

* .move_range("数据区域",rows=,cols=)：正整数为向下或向右、负整数为向左或向上；

向左移动两列，向下移动两行 sheet.move_range("C1:D4",rows=2,cols=-1)

演示效果如下：



	A	B	C	D
1	你好啊	123	张三	1
2	你好啊	321	李四	2
3			王五	3
4			赵六	4

	A	B
1	你好啊	123
2	你好啊	321
3	张三	1
4	李四	2
5	王五	3
6	赵六	4

⑦ .create_sheet()：创建新的 sheet 表格

* .create_sheet("新的 sheet 名")：创建一个新的 sheet 表；

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active
print(sheet)
workbook.create_sheet("我是一个新的 sheet")
print(workbook.sheetnames)
workbook.save(filename = "test.xlsx")
```

结果如下：

```
In [93]: workbook = load_workbook(filename = "test.xlsx")
         sheet = workbook.active
         print(sheet)
         workbook.create_sheet("我是一个新的sheet")
         print(workbook.sheetnames)
         workbook.save(filename = "test.xlsx")

         <Worksheet "Sheet4">
         ['Sheet1', 'Sheet2', 'Sheet3', 'Sheet4', '我是一个新的sheet']
```

⑧ .remove()：删除某个 sheet 表

* .remove("sheet 名")：删除某个 sheet 表；

```
workbook = load_workbook(filename = "test.xlsx")
sheet = workbook.active print(workbook.sheetnames)
# 这个相当于激活的这个 sheet 表，激活状态下，才可以操作；
sheet = workbook['我是一个新的 sheet']
print(sheet)
workbook.remove(sheet)
print(workbook.sheetnames)
workbook.save(filename = "test.xlsx")
```

结果如下：

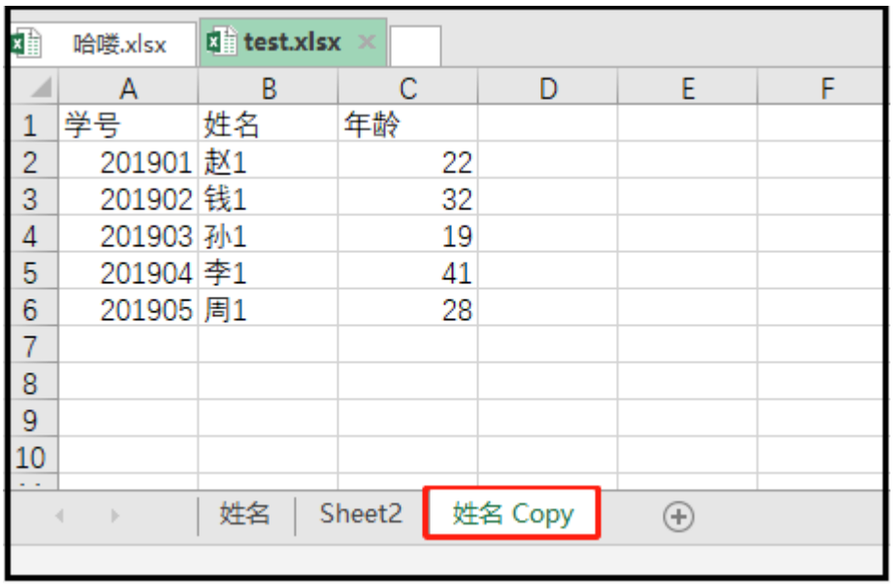
```
In [105]: workbook = load_workbook(filename = "test.xlsx")
          sheet = workbook.active
          print(workbook.sheetnames)
          # 这个相当于激活的这个sheet表，激活状态下，才可以操作；
          sheet = workbook['我是一个新的sheet']
          print(sheet)
          workbook.remove(sheet)
          print(workbook.sheetnames)
          workbook.save(filename = "test.xlsx")

          ['Sheet1', 'Sheet2', 'Sheet3', 'Sheet4', '我是一个新的sheet']
          <Worksheet "我是一个新的sheet">
          ['Sheet1', 'Sheet2', 'Sheet3', 'Sheet4']
```

⑨ .copy_worksheet()：复制一个 sheet 表到另外一张 excel 表

* 这个操作的实质，就是复制某个 excel 表中的 sheet 表，然后将文件存储到另外一张 excel 表中；

```
workbook = load_workbook(filename = "a.xlsx")
sheet = workbook.active
print("a.xlsx 中有这几个 sheet 表",workbook.sheetnames)
sheet = workbook['姓名']
workbook.copy_worksheet(sheet)
workbook.save(filename = "test.xlsx")
```

结果如下：

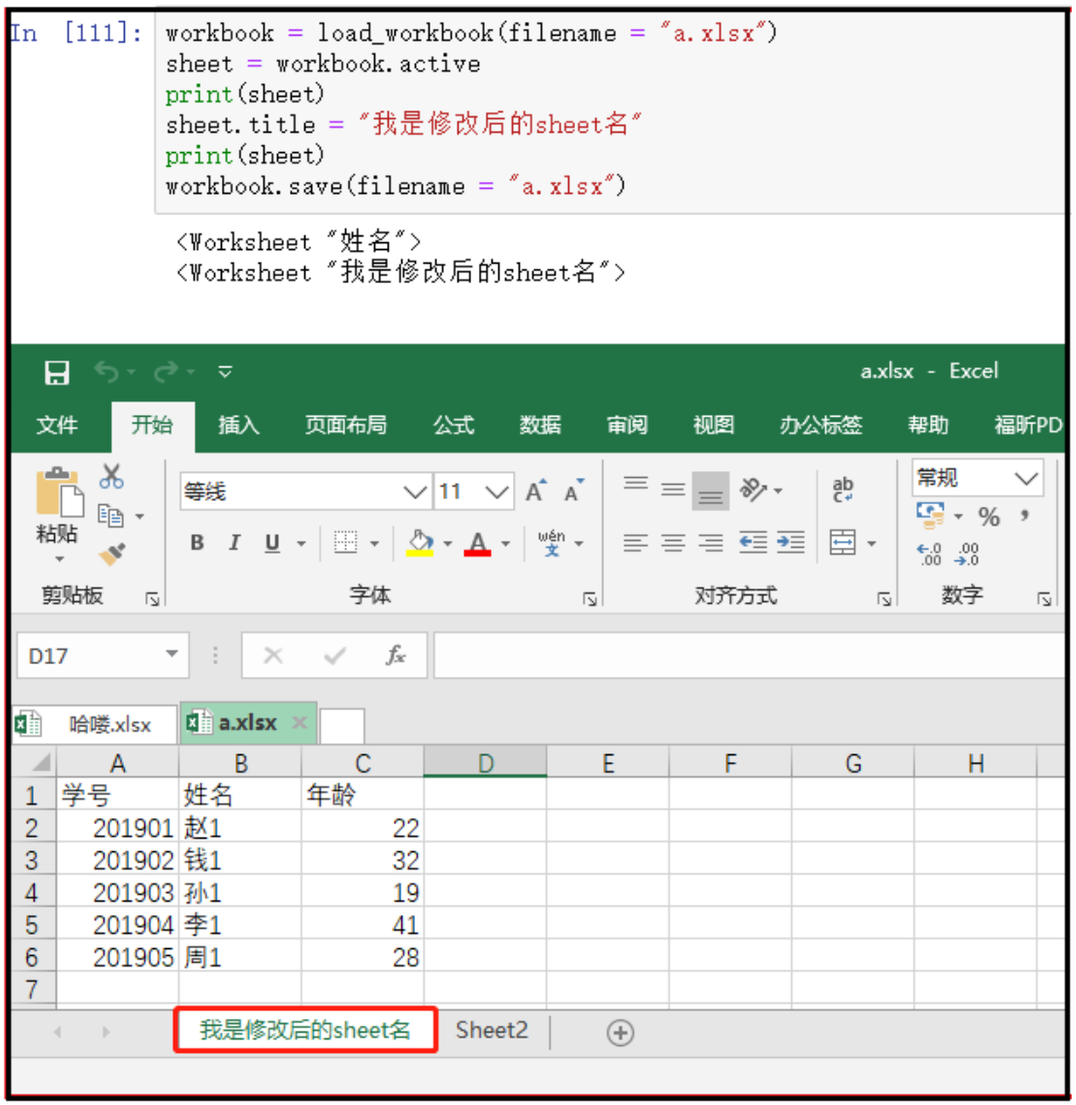


⑩ sheet.title：修改 sheet 表的名称

* .title = "新的 sheet 表名"

```
workbook = load_workbook(filename = "a.xlsx")
sheet = workbook.active
print(sheet)
sheet.title = "我是修改后的 sheet 名"
print(sheet)
```

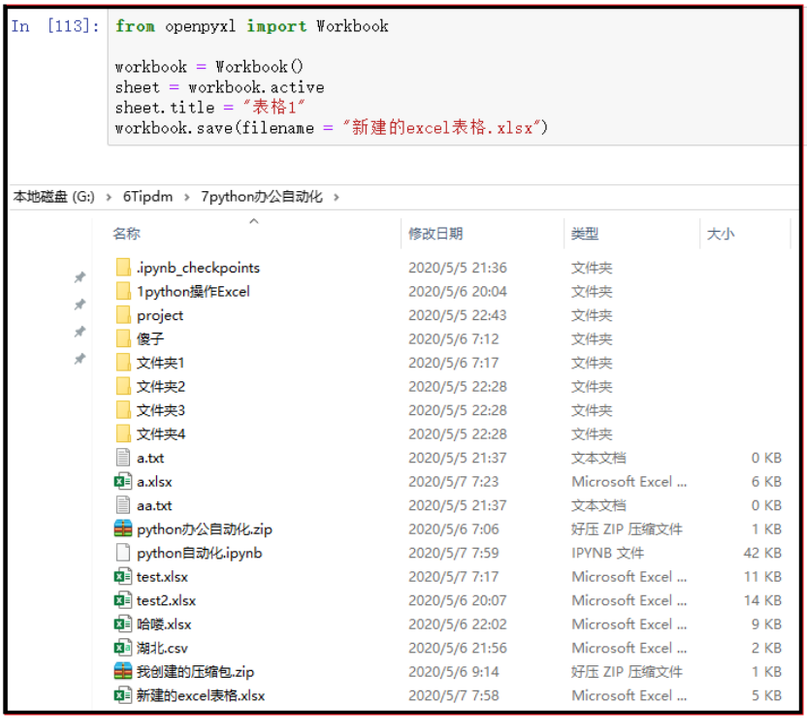
结果如下：



⑪ 创建新的 excel 表格文件

```
from openpyxl import Workbook
workbook = Workbook()
sheet = workbook.active
sheet.title = "表格 1"
workbook.save(filename = "新建的 excel 表格")
```

结果如下：



⑫ sheet.freeze_panes：冻结窗口

* .freeze_panes = "单元格"

```
workbook = load_workbook(filename = "花园.xlsx")
sheet = workbook.active print(sheet) sheet.freeze_panes = "C3"
workbook.save(filename = "花园.xlsx")
"""
冻结窗口以后，你可以打开源文件，进行检验；
"""
```

结果如下：

```
In [115]: workbook = load_workbook(filename = "花园.xlsx")
          sheet = workbook.active
          print(sheet)
          sheet.freeze_panes = "C3"
          workbook.save(filename = "花园.xlsx")

          <Worksheet "Sheet1">
```

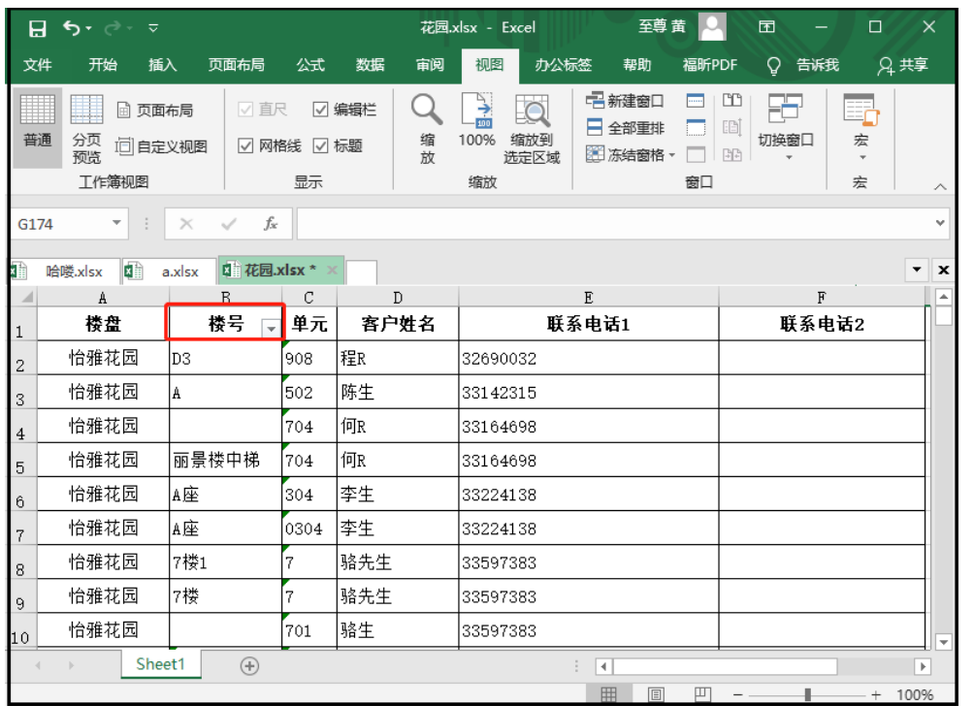
⑬ sheet.auto_filter.ref：给表格添加“筛选器”

* .auto_filter.ref = sheet.dimension 给所有字段添加筛选器；

* .auto_filter.ref = "A1" 给 A1 这个格子添加“筛选器”，就是给第一列添加“筛选器”；

```
workbook = load_workbook(filename = "花园.xlsx")
sheet = workbook.active
print(sheet)
sheet.auto_filter.ref = sheet["A1"]
workbook.save(filename = "花园.xlsx")
```

结果如下：



楼盘	楼号	单元	客户姓名	联系电话1	联系电话2
怡雅花园	D3	908	程R	32690032	
怡雅花园	A	502	陈生	33142315	
怡雅花园		704	何R	33164698	
怡雅花园	丽景楼中梯	704	何R	33164698	
怡雅花园	A座	304	李生	33224138	
怡雅花园	A座	0304	李生	33224138	
怡雅花园	7楼1	7	骆先生	33597383	
怡雅花园	7楼	7	骆先生	33597383	
怡雅花园		701	骆生	33597383	

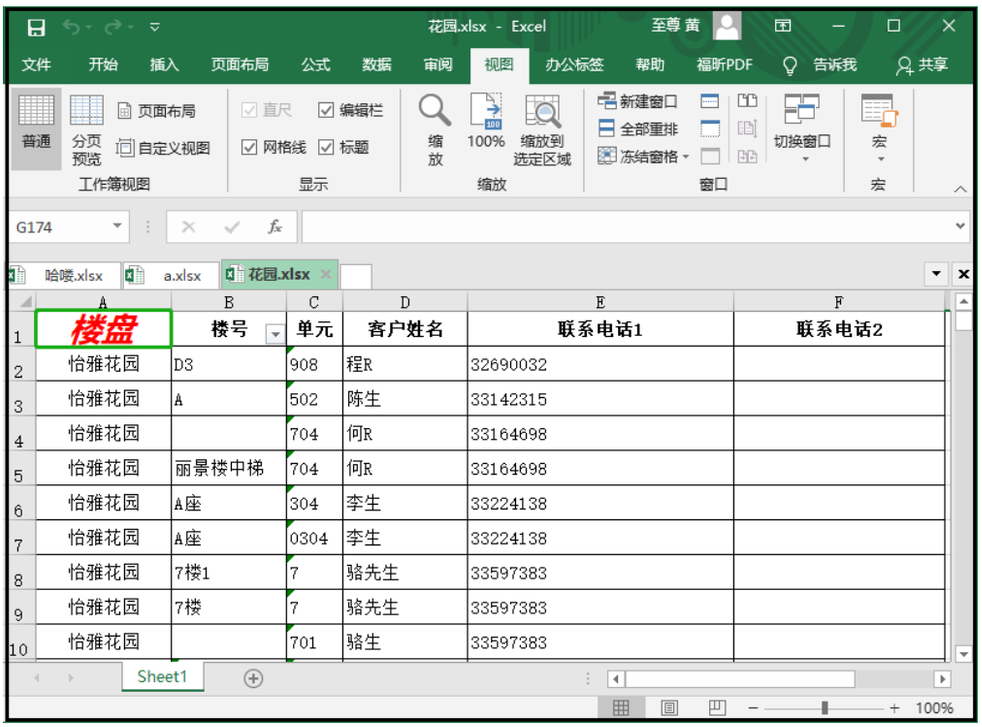
3、批量调整字体和样式

1）修改字体样式

* Font(name=字体名称,size=字体大小,bold=是否加粗,italic=是否斜体,color=字体颜色)

```
from openpyxl.styles import Font
from openpyxl import load_workbook
workbook = load_workbook(filename="花园.xlsx")
sheet = workbook.active
cell = sheet["A1"]
font = Font(name="微软雅黑",size=20,bold=True,italic=True,color="FF0000")
cell.font = font
workbook.save(filename = "花园.xlsx")
"""
这个 color 是 RGB 的 16 进制表示，自己下去百度学习；
"""
```

结果如下：



楼盘	楼号	单元	客户姓名	联系电话1	联系电话2
怡雅花园	D3	908	程R	32690032	
怡雅花园	A	502	陈生	33142315	
怡雅花园		704	何R	33164698	
怡雅花园	丽景楼中梯	704	何R	33164698	
怡雅花园	A座	304	李生	33224138	
怡雅花园	A座	0304	李生	33224138	
怡雅花园	7楼1	7	骆先生	33597383	
怡雅花园	7楼	7	骆先生	33597383	
怡雅花园		701	骆生	33597383	

2）获取表格中格子的字体样式

```
from openpyxl.styles import Font
from openpyxl import load_workbook
workbook = load_workbook(filename="花园.xlsx")
sheet = workbook.active
cell = sheet["A2"]
font = cell.font
print(font.name, font.size, font.bold, font.italic, font.color)
```

结果如下：

```
In [127]: from openpyxl.styles import Font
          from openpyxl import load_workbook

          workbook = load_workbook(filename="花园.xlsx")
          sheet = workbook.active
          cell = sheet["A2"]
          font = cell.font
          print(font.name, font.size, font.bold, font.italic, font.color)

          宋体 11.0 False False <openpyxl.styles.colors.Color object>
          Parameters:
          rgb=None, indexed=None, auto=None, theme=1, tint=0.0, type='theme'
```

3）设置对齐样式

* Alignment(horizontal=水平对齐模式,vertical=垂直对齐模式,text_rotation=旋转角度,wrap_text=是否自动换行)

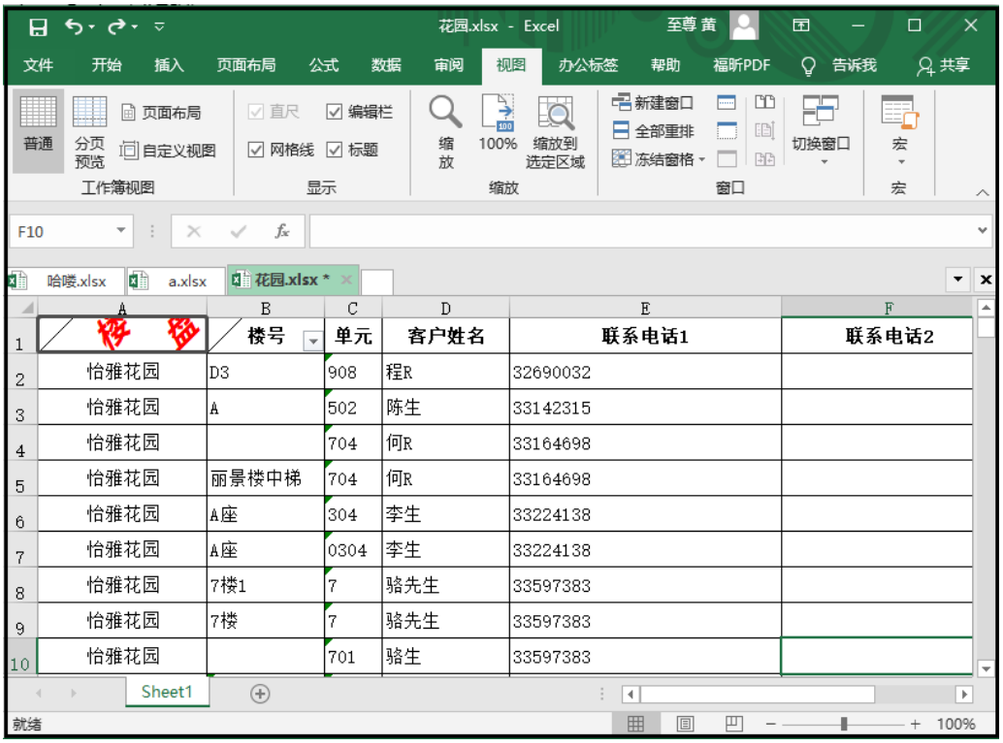
* 水平对齐：‘distributed'，‘justify'，‘center'，‘leftfill'， ‘centerContinuous'，‘right，‘general';

* 垂直对齐：‘bottom'，‘distributed'，‘justify'，‘center'，‘top';

```
from openpyxl.styles import Alignment
from openpyxl import load_workbook
workbook = load_workbook(filename="花园.xlsx")
sheet = workbook.active
cell = sheet["A1"]
```

```
alignment =
Alignment(horizontal="center",vertical="center",text_rotation=45,wrap_text=True)
cell.alignment = alignment
workbook.save(filename = "花园.xlsx")
```

结果如下：



楼盘	楼号	单元	客户姓名	联系电话1	联系电话2
怡雅花园	D3	908	程R	32690032	
怡雅花园	A	502	陈生	33142315	
怡雅花园		704	何R	33164698	
怡雅花园	丽景楼中梯	704	何R	33164698	
怡雅花园	A座	304	李生	33224138	
怡雅花园	A座	0304	李生	33224138	
怡雅花园	7楼1	7	骆先生	33597383	
怡雅花园	7楼	7	骆先生	33597383	
怡雅花园		701	骆生	33597383	

4）设置边框样式

* Side(style=边线样式，color=边线颜色)

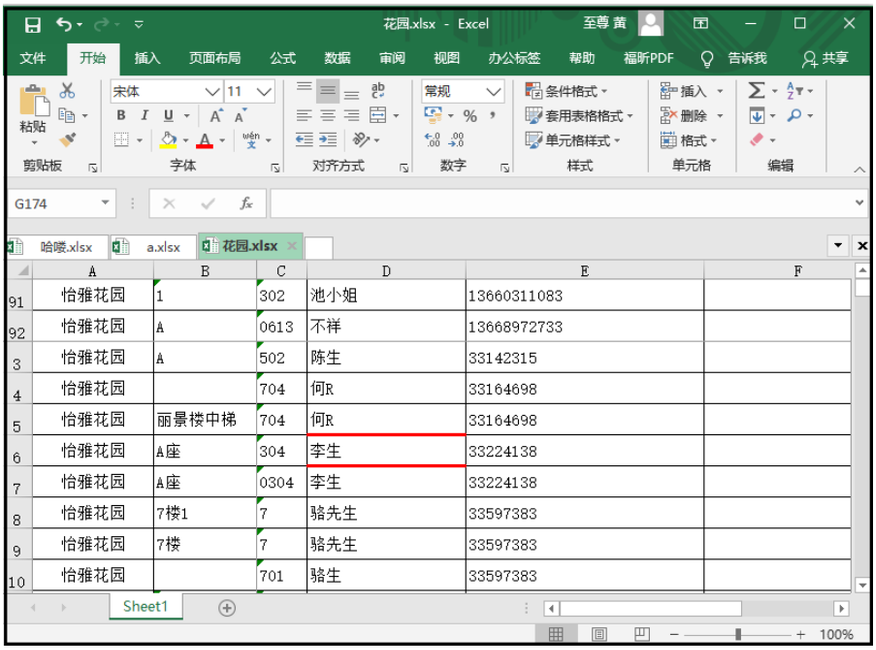
* Border(left=左边线样式，right=右边线样式，top=上边线样式，bottom=下边线样式)

* style 参数的种类： 'double, 'mediumDashDotDot', 'slantDashDot', 'dashDotDot','dotted','hair', 'mediumDashed, 'dashed', 'dashDot', 'thin', 'mediumDashDot','medium', 'thick'

```
from openpyxl.styles import Side,Border
from openpyxl import load_workbook
workbook = load_workbook(filename="花园.xlsx")
sheet = workbook.active
cell = sheet["D6"]
```

```
side1 = Side(style="thin",color="FF0000")
side2 = Side(style="thick",color="FFFF0000")
border = Border(left=side1,right=side1,top=side2,bottom=side2)
cell.border = border
workbook.save(filename = "花园.xlsx")
```

结果如下：



	A	B	C	D	E	F
91	怡雅花园	1	302	池小姐	13660311083	
92	怡雅花园	A	0613	不祥	13668972733	
3	怡雅花园	A	502	陈生	33142315	
4	怡雅花园		704	何R	33164698	
5	怡雅花园	丽景楼中梯	704	何R	33164698	
6	怡雅花园	A座	304	李生	33224138	
7	怡雅花园	A座	0304	李生	33224138	
8	怡雅花园	7楼1	7	骆先生	33597383	
9	怡雅花园	7楼	7	骆先生	33597383	
10	怡雅花园		701	骆生	33597383	

5）设置填充样式

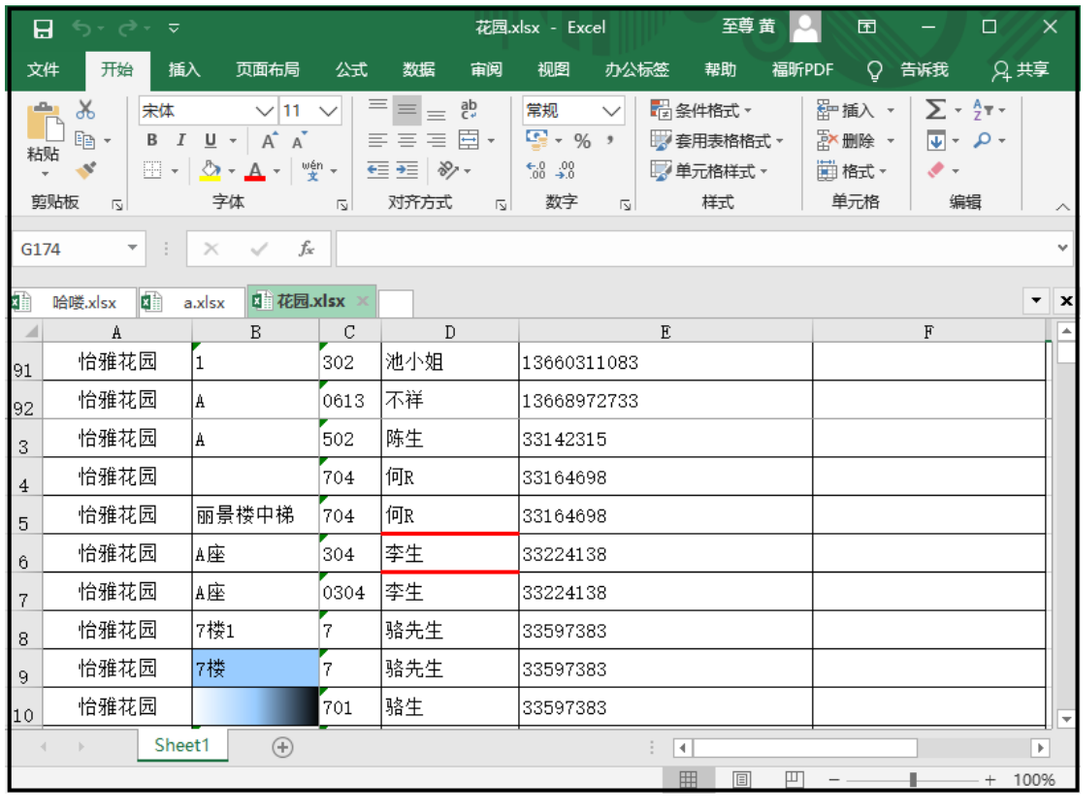
* PatternFill(fill_type=填充样式，fgColor=填充颜色）

* GradientFill(stop=(渐变颜色 1，渐变颜色 2……))

```
from openpyxl.styles import PatternFill,GradientFill
from openpyxl import load_workbook
workbook = load_workbook(filename="花园.xlsx")
sheet = workbook.active
cell_b9 = sheet["B9"]
pattern_fill = PatternFill(fill_type="solid",fgColor="99ccff")
cell_b9.fill = pattern_fill
cell_b10 = sheet["B10"]
```

```
gradient_fill = GradientFill(stop=("FFFFFF","99ccff","000000"))
cell_b10.fill = gradient_fill
workbook.save(filename = "花园.xlsx")
```

结果如下：



	A	B	C	D	E	F
91	怡雅花园	1	302	池小姐	13660311083	
92	怡雅花园	A	0613	不祥	13668972733	
3	怡雅花园	A	502	陈生	33142315	
4	怡雅花园		704	何R	33164698	
5	怡雅花园	丽景楼中梯	704	何R	33164698	
6	怡雅花园	A座	304	李生	33224138	
7	怡雅花园	A座	0304	李生	33224138	
8	怡雅花园	7楼1	7	骆先生	33597383	
9	怡雅花园	7楼	7	骆先生	33597383	
10	怡雅花园		701	骆生	33597383	

6）设置行高和列宽

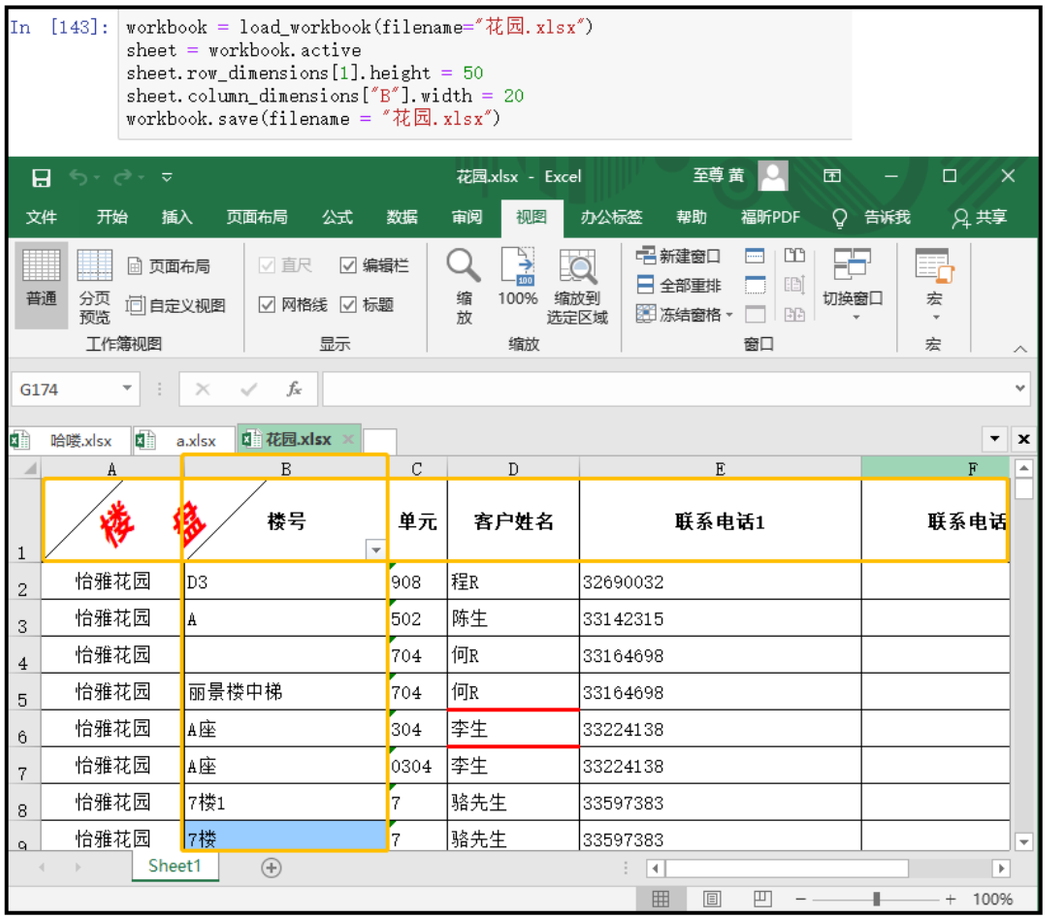
* .row_dimensions[行编号].height = 行高

* .column_dimensions[列编号].width = 列宽

```
workbook = load_workbook(filename="花园.xlsx")
sheet = workbook.active
# 设置第 1 行的高度
sheet.row_dimensions[1].height = 50
# 设置 B 列的宽度
sheet.column_dimensions["B"].width = 20
workbook.save(filename = "花园.xlsx")
```

```
"""
sheet.row_dimensions.height = 50
sheet.column_dimensions.width = 30
这两句代码，是将整个表的行高设置为 50，列宽设置为 30；
"""
```

结果如下：



7）合并单元格

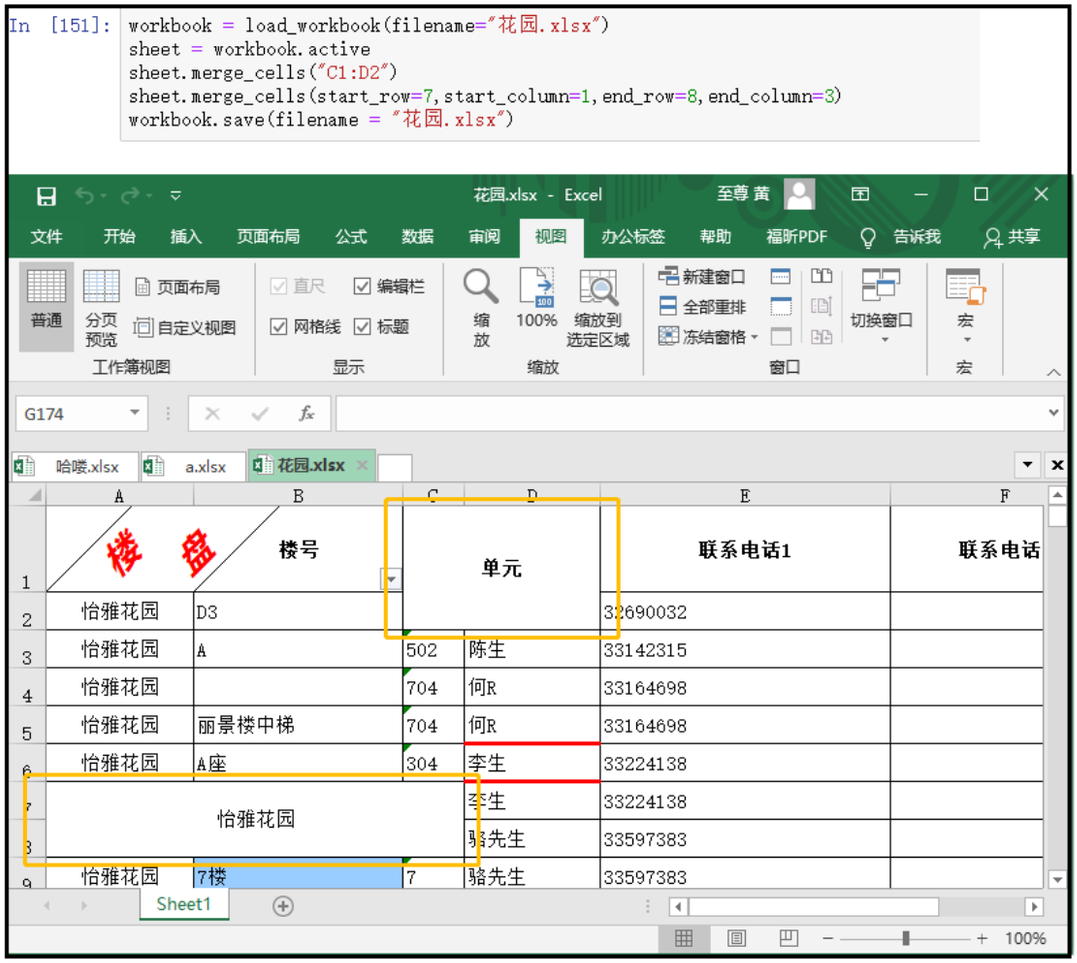
* .merge_cells(待合并的格子编号)

* .merge_cells(start_row=起始行号，start_column=起始列号，end_row=结束行号，end_column=结束列号)

```
workbook = load_workbook(filename="花园.xlsx")
```

```
sheet = workbook.active sheet.merge_cells("C1:D2")
sheet.merge_cells(start_row=7,start_column=1,end_row=8,end_column=3)
workbook.save(filename = "花园.xlsx")
```

结果如下：



当然，也有“取消合并单元格”，用法一致。

* .unmerge_cells(待合并的格子编号)

* .unmerge_cells(start_row=起始行号，start_column=起始列号，end_row=结束行号，end_column=结束列号)

章节二：python 使用 PyPDF2 和 pdfplumber 操作 pdf

1、PyPDF2 和 pdfplumber 库介绍

* PyPDF2 官网：https://pythonhosted.org/PyPDF2/

* PyPDF2 可以更好的读取、写入、分割、合并 PDF 文件；

* pdfplumber 官网：https://github.com/jsvine/pdfplumber

* pdfplumber 可以更好地读取 PDF 文件内容和提取 PDF 中的表格；

* 这两个库不属于 python 标准库，都需要单独安装；

2、python 提取 PDF 文字内容

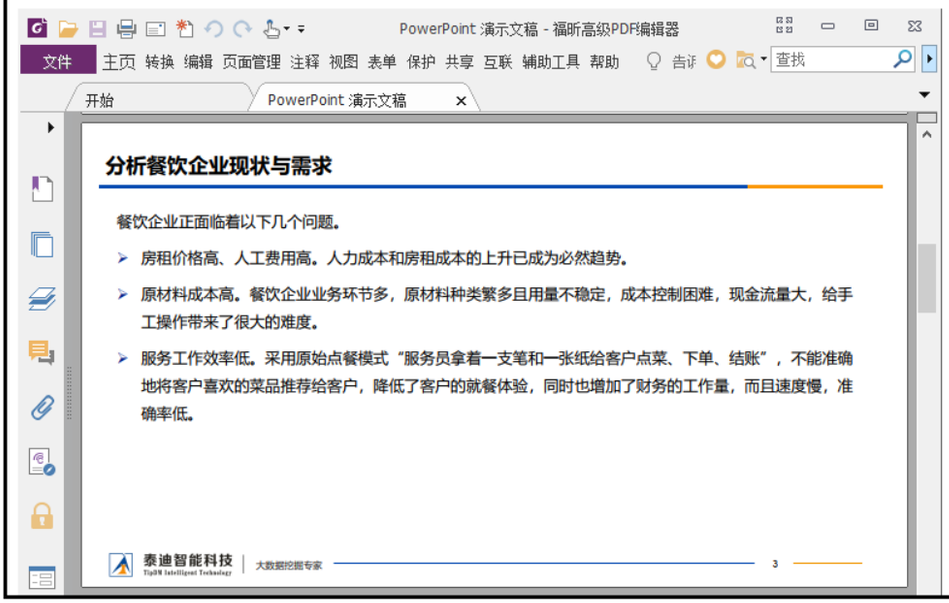
1）利用 pdfplumber 提取文字

```
import PyPDF2
import pdfplumber


with pdfplumber.open("餐饮企业综合分析.pdf") as p:
    page = p.pages[2]
    print(page.extract_text())
```

结果如下：

```
In [9]: import PyPDF2
        import pdfplumber

        with pdfplumber.open("餐饮企业综合分析.pdf") as p:
            page = p.pages[2]
            print(page.extract_text())
```

```
分析餐饮企业现状与需求
餐饮企业正面临着以下几个问题。
房租价格高、人工费用高。人力成本和房租成本的上升已成为必然趋势。
□
原材料成本高。餐饮企业业务环节多，原材料种类繁多且用量不稳定，成本控制困难，现金流量大，给手
□
工操作带来了很大的难度。
服务工作效率低。采用原始点餐模式“服务员拿着一支笔和一张纸给客户点菜、下单、结账”，不能准确
□
地将客户喜欢的菜品推荐给客户，降低了客户的就餐体验，同时也增加了财务的工作量，而且速度慢，准
确率低。
大数据挖掘专家 3
```



2）利用 pdfplumber 提取表格并写入 excel

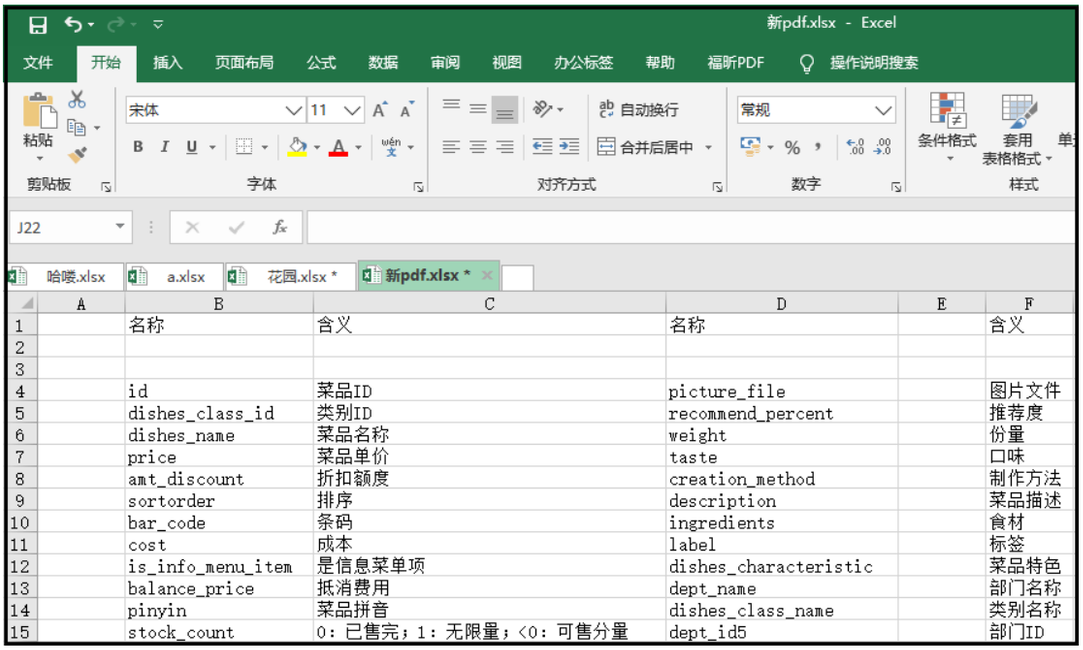
* extract_table()：如果一页有一个表格；

* extract_tables()：如果一页有多个表格；

```
import PyPDF2
import pdfplumber
from openpyxl import Workbook
with pdfplumber.open("餐饮企业综合分析.pdf") as p:
    page = p.pages[4]
    table = page.extract_table()
    print(table)
    workbook = Workbook()
```

```
    sheet = workbook.active
    for row in table:
            sheet.append(row)
    workbook.save(filename = "新 pdf.xlsx")
```

结果如下：



	A	B	C	D	E	F
1		名称	含义	名称		含义
2						
3						
4		id	菜品ID	picture_file		图片文件
5		dishes_class_id	类别ID	recommend_percent		推荐度
6		dishes_name	菜品名称	weight		份量
7		price	菜品单价	taste		口味
8		amt_discount	折扣额度	creation_method		制作方法
9		sortorder	排序	description		菜品描述
10		bar_code	条码	ingredients		食材
11		cost	成本	label		标签
12		is_info_menu_item	是信息菜单项	dishes_characteristic		菜品特色
13		balance_price	抵消费用	dept_name		部门名称
14		pinyin	菜品拼音	dishes_class_name		类别名称
15		stock_count	0：已售完；1：无限量；<0：可售分量	dept_id5		部门ID

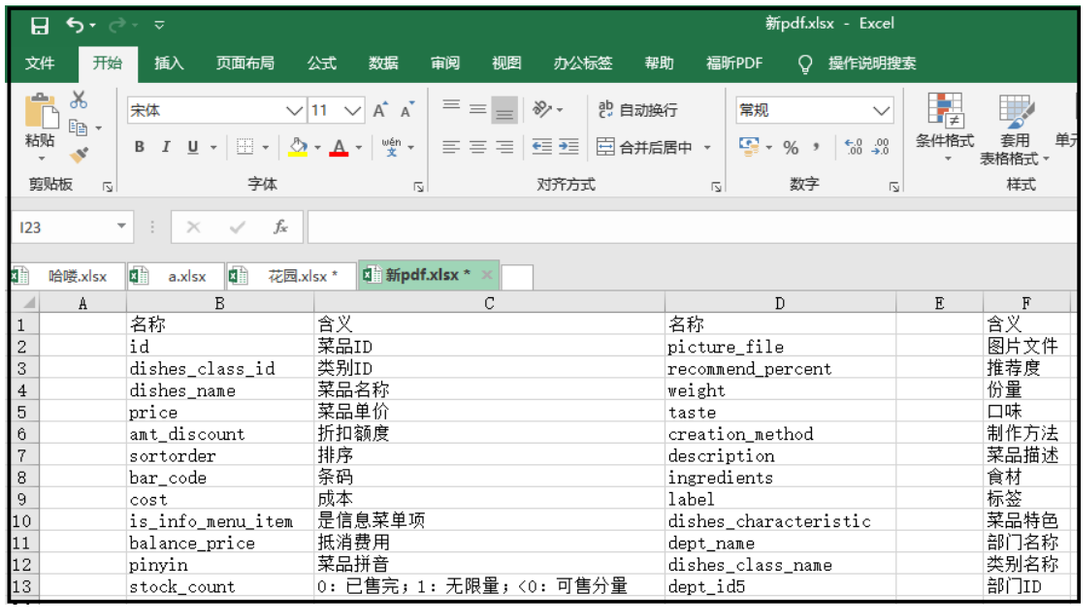
缺陷：可以看到，这里提取出来的表格有很多空行，怎么去掉这些空行呢？

判断：将列表中每个元素都连接成一个字符串，如果还是一个空字符串那么肯定就是空行。

```
import PyPDF2
import pdfplumber
from openpyxl import Workbook
with pdfplumber.open("餐饮企业综合分析.pdf") as p:
    page = p.pages[4]
    table = page.extract_table()
    print(table)
    workbook = Workbook()
    sheet = workbook.active
    for row in table:
        if not "".join([str(i) for i in row]) == "":
```

```
            sheet.append(row)
    workbook.save(filename = "新 pdf.xlsx")
```

结果如下：



	A	B	C	D	E	F
1		名称	含义	名称		含义
2		id	菜品ID	picture_file		图片文件
3		dishes_class_id	类别ID	recommend_percent		推荐度
4		dishes_name	菜品名称	weight		份量
5		price	菜品单价	taste		口味
6		amt_discount	折扣额度	creation_method		制作方法
7		sortorder	排序	description		菜品描述
8		bar_code	条码	ingredients		食材
9		cost	成本	label		标签
10		is_info_menu_item	是信息菜单项	dishes_characteristic		菜品特色
11		balance_price	抵消费用	dept_name		部门名称
12		pinyin	菜品拼音	dishes_class_name		类别名称
13		stock_count	0：已售完；1：无限量；<0：可售分量	dept_id5		部门ID

3、PDF 合并及页面的排序和旋转

1）分割及合并 pdf

① 合并 pdf

首先，我们有如下几个文件，可以发现这里共有三个 PDF 文件需要我们合并。同时可以发现他们的文件名都是有规律的(如果文件名，没有先后顺序，我们合并起来就没有意义了。)

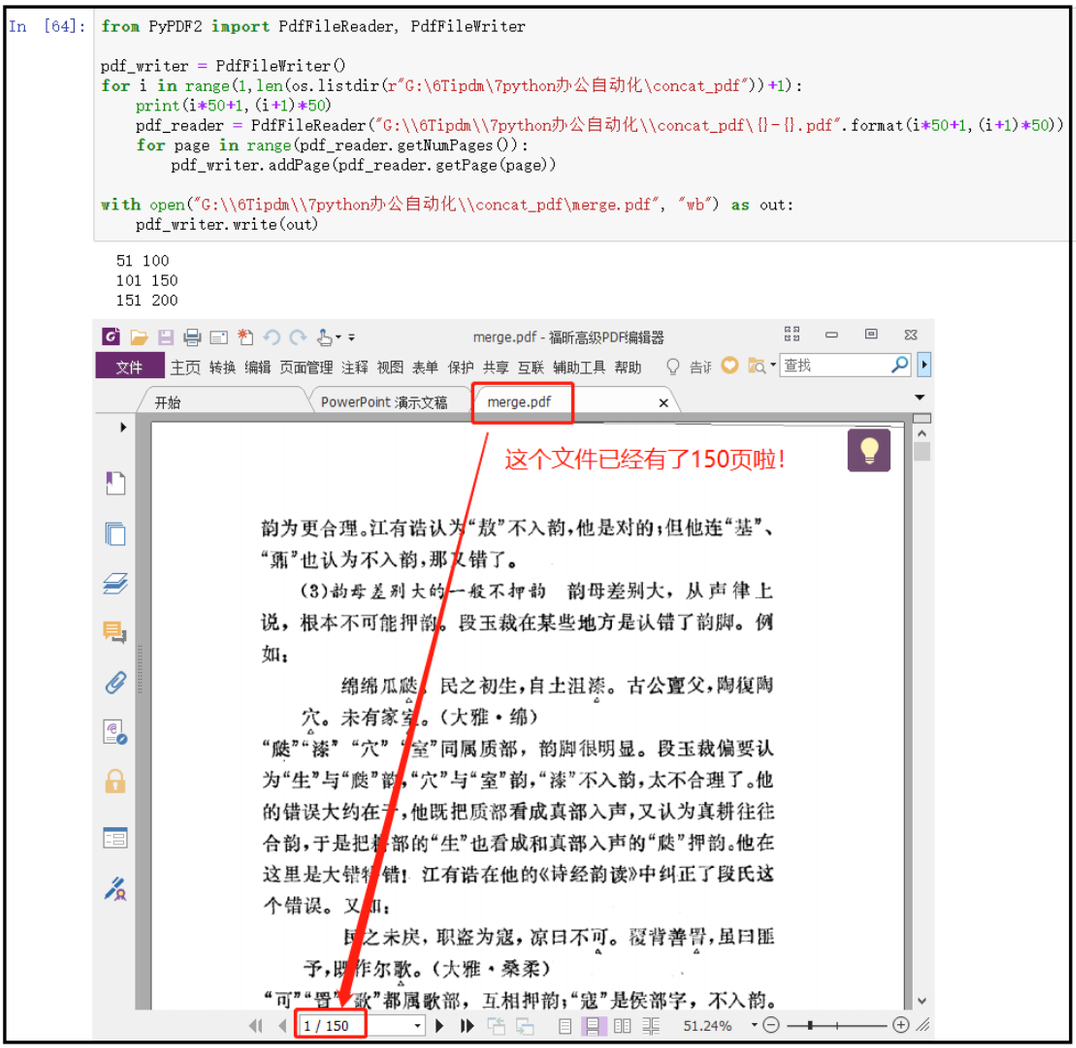
名称	修改日期	类型	大小
51-100.pdf	2020/5/3 18:08	Foxit PhantomP...	2,781 KB
101-150.pdf	2020/5/3 18:09	Foxit PhantomP...	1,855 KB
151-200.pdf	2020/5/4 6:56	Foxit PhantomP...	1,237 KB

代码如下：

```
from PyPDF2 import PdfFileReader, PdfFileWriter
pdf_writer = PdfFileWriter()
for i in range(1,len(os.listdir(r"G:\6Tipdm\7python 办公自动化\concat_pdf"))+1):
```

```
        print(i*50+1,(i+1)*50)
        pdf_reader = PdfFileReader("G:\\6Tipdm\\7python 办公自动化\\concat_pdf\{}-{}.pdf".format(i*50+1,(i+1)*50))
        for page in range(pdf_reader.getNumPages()):
            pdf_writer.addPage(pdf_reader.getPage(page))
with open("G:\\6Tipdm\\7python 办公自动化\\concat_pdf\merge.pdf", "wb") as out:
        pdf_writer.write(out)
```

结果如下：



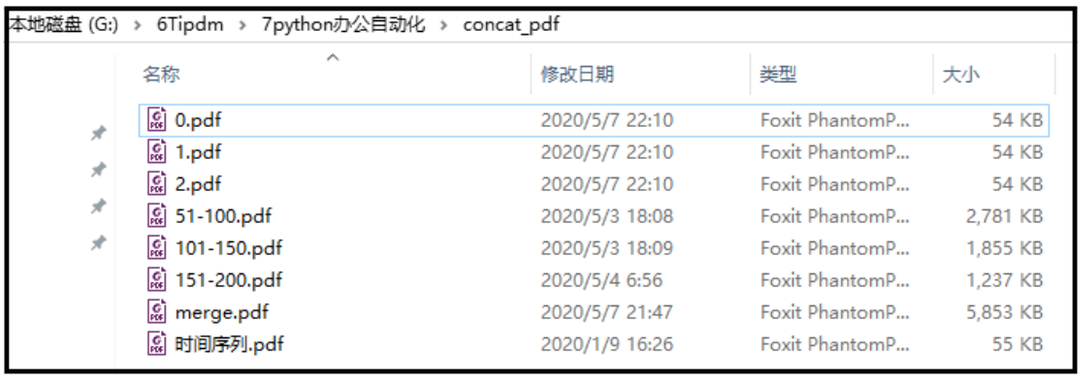
② 拆分 pdf

这里有一个“时间序列.pdf”的文件，共 3 页，我们将其每一页存为一个 PDF 文件。

代码如下：

```
from PyPDF2 import PdfFileReader, PdfFileWriter
pdf_reader = PdfFileReader(r"G:\6Tipdm\7python 办公自动化\concat_pdf\时间序列.pdf")
for page in range(pdf_reader.getNumPages()):
    pdf_writer = PdfFileWriter()
    pdf_writer.addPage(pdf_reader.getPage(page))
    with open(f"G:\\6Tipdm\\7python 办公自动化\\concat_pdf\\{page}.pdf", "wb") as out:
        pdf_writer.write(out)
```

结果如下：



2）旋转及排序 pdf

① 旋转 pdf

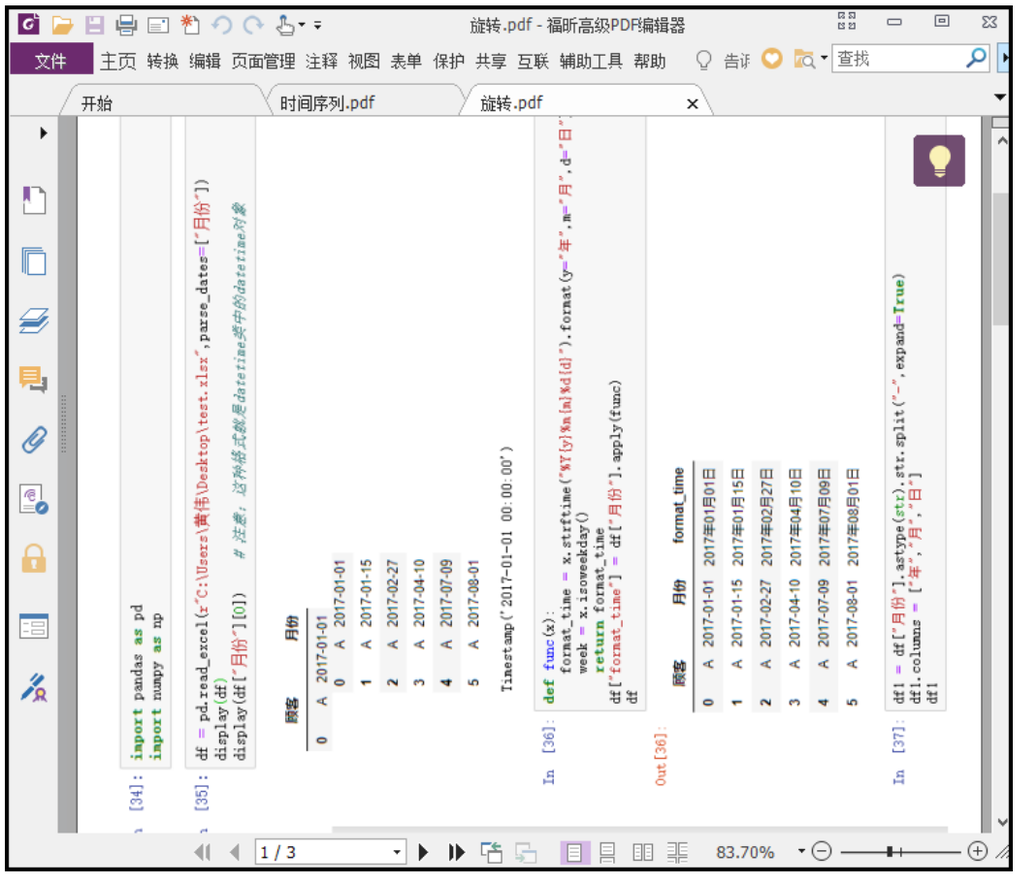
* .rotateClockwise(90 的倍数)：顺时针旋转 90 度

* .rotateCounterClockwise(90 的倍数)：逆时针旋转 90 度

```
from PyPDF2 import PdfFileReader, PdfFileWriter
```

```
pdf_reader = PdfFileReader(r"G:\6Tipdm\7python 办公自动化\concat_pdf\时间序列.pdf")
pdf_writer = PdfFileWriter()
for page in range(pdf_reader.getNumPages()):
    if page % 2 == 0:
        rotation_page = pdf_reader.getPage(page).rotateCounterClockwise(90)
    else:
        rotation_page = pdf_reader.getPage(page).rotateClockwise(90)
    pdf_writer.addPage(rotation_page)
    with open("G:\\6Tipdm\\7python 办公自动化\\concat_pdf\\旋转.pdf", "wb") as out:
        pdf_writer.write(out)
"""
上述代码中，我们循环遍历了这个 pdf，对于偶数页我们逆时针旋转 90°，对于奇数页我们顺时针旋转 90°；  注意：旋转的角度只能是 90 的倍数；
"""
```

其中一页效果展示如下：



② 排序 pdf

需求：我们有一个 PDF 文件，我们需要倒序排列，应该怎么做呢？

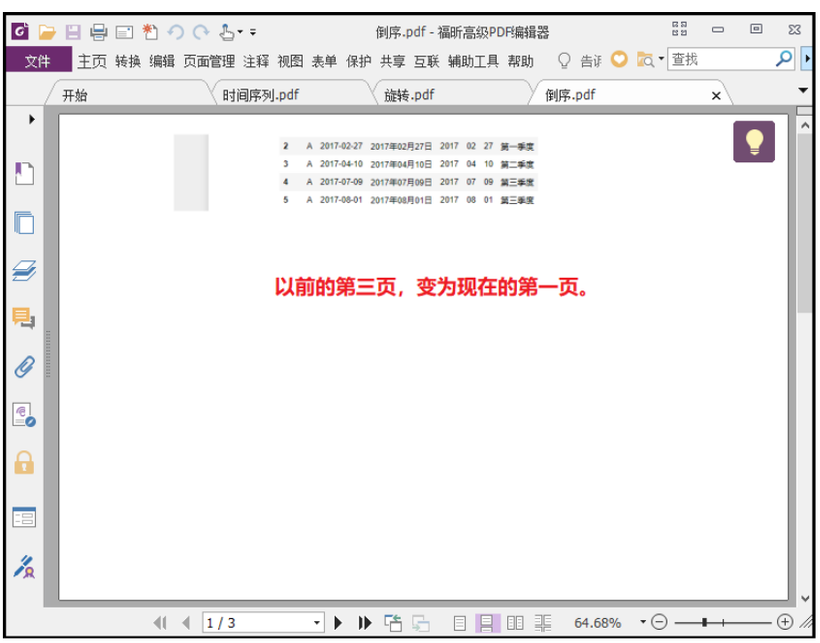
首先，我们来看 python 中，怎么倒叙打印一串数字，如下图所示。

```
In [75]: for i in range(5,-1,-1):
             print(i)

         5
         4
         3
         2
         1
         0
```

那么倒序排列一个 pdf，思路同上，代码如下：

```
from PyPDF2 import PdfFileReader, PdfFileWriter
pdf_reader = PdfFileReader(r"G:\6Tipdm\7python 办公自动化\concat_pdf\时间序列.pdf")
pdf_writer = PdfFileWriter()
for page in range(pdf_reader.getNumPages()-1, -1, -1):
    pdf_writer.addPage(pdf_reader.getPage(page))
with open("G:\\6Tipdm\\7python 办公自动化\\concat_pdf\\倒序.pdf", "wb") as out:
    pdf_writer.write(out)
```

结果如下：



4、pdf 批量加水印及加密、解密

1）批量加水印

```
from PyPDF2 import PdfFileReader, PdfFileWriter
from copy import copy

water = PdfFileReader(r"G:\6Tipdm\7python 办公自动化\concat_pdf\水印.pdf")
water_page = water.getPage(0)

pdf_reader = PdfFileReader(r"G:\6Tipdm\7python 办公自动化\concat_pdf\aa.pdf")
pdf_writer = PdfFileWriter()

for page in range(pdf_reader.getNumPages()):
    my_page = pdf_reader.getPage(page)
    new_page = copy(water_page)
    new_page.mergePage(my_page)
    pdf_writer.addPage(new_page)
with open("G:\\6Tipdm\\7python 办公自动化\\concat_pdf\\添加水印后的 aa.pdf", "wb") as
     out: pdf_writer.write(out)
"""
这里有一点需要注意：进行 pdf 合并的时候，我们希望“水印”在下面，文字在上面，因
     此
是“水印”.mergePage(“图片页”)
"""
```

结果如下：

2）批量加密、解密

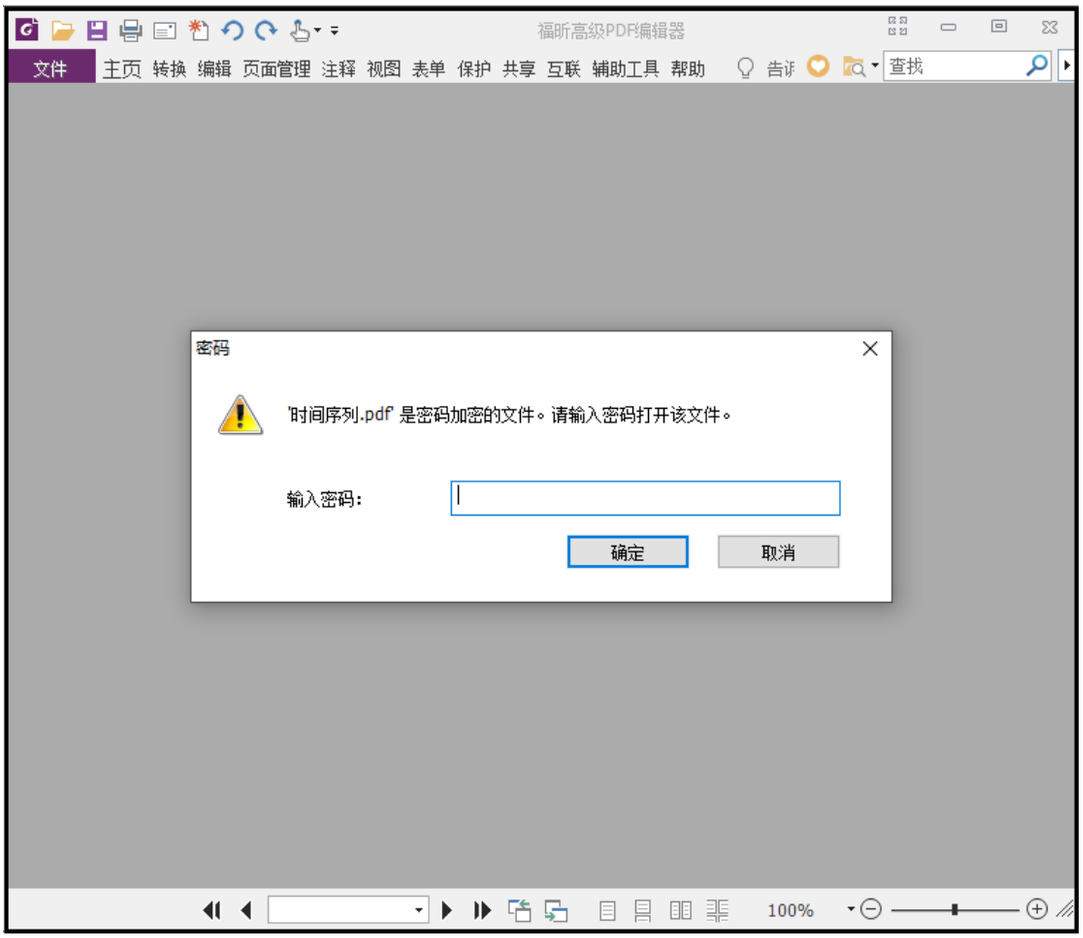
* 这里所说的“解密”，是在知道 pdf 的密码下，去打开 pdf，而不是暴力破解；

① 加密 pdf

```
from PyPDF2 import PdfFileReader, PdfFileWriter
pdf_reader = PdfFileReader(r"G:\6Tipdm\7python 办公自动化\concat_pdf\时间序列.pdf")
pdf_writer = PdfFileWriter()
for page in range(pdf_reader.getNumPages()):
    pdf_writer.addPage(pdf_reader.getPage(page))
# 添加密码
pdf_writer.encrypt("a123456")
```

```
with open("G:\\6Tipdm\\7python 办公自动化\\concat_pdf\\时间序列.pdf", "wb") as out:
    pdf_writer.write(out)
```

结果如下：

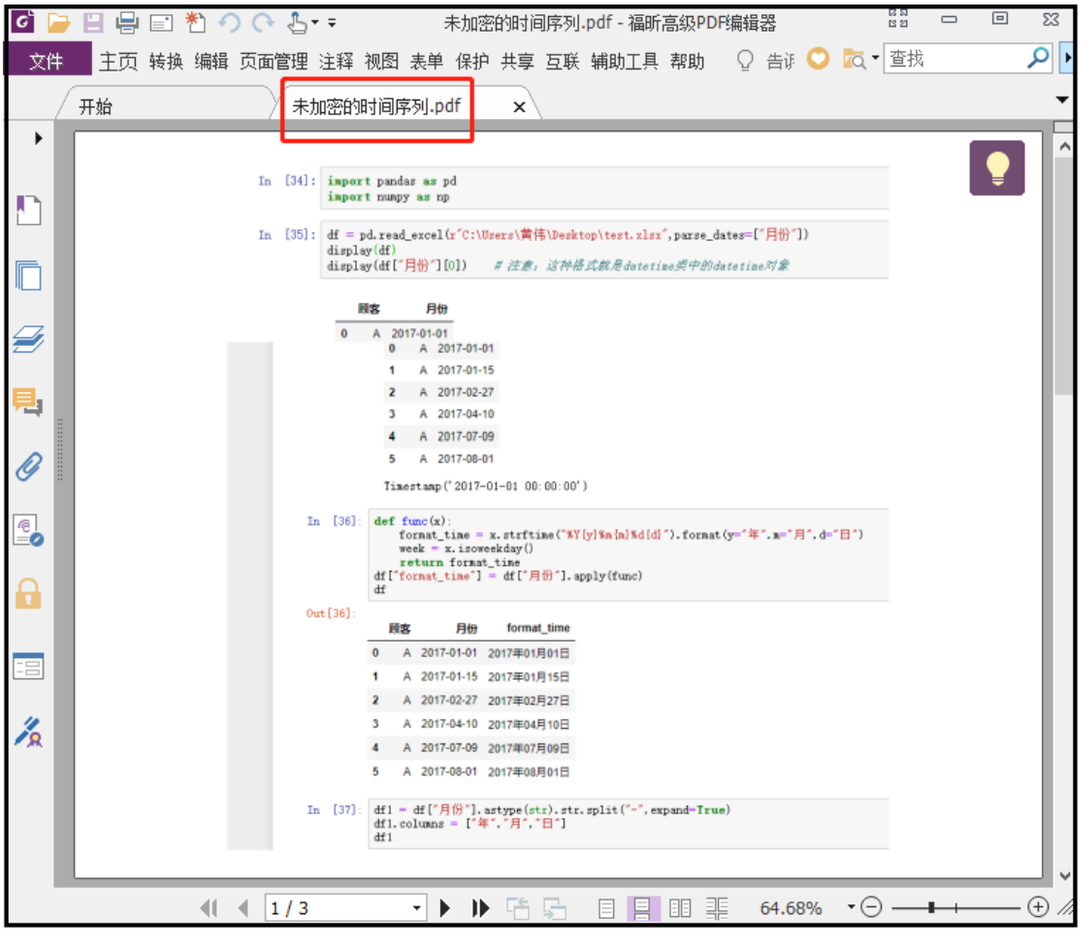


② 解密 pdf 并保存为未加密的 pdf

```
from PyPDF2 import PdfFileReader, PdfFileWriter
pdf_reader = PdfFileReader(r"G:\6Tipdm\7python 办公自动化\concat_pdf\时间序列.pdf")
# 解密
pdf pdf_reader.decrypt("a123456")
pdf_writer = PdfFileWriter()
for page in range(pdf_reader.getNumPages()):
    pdf_writer.addPage(pdf_reader.getPage(page))
```

```
with open("G:\\6Tipdm\\7python 办公自动化\\concat_pdf\\未加密的时间序列.pdf", "wb") as out:
    pdf_writer.write(out)
```

结果如下：



章节三：python 使用 python-docx 操作 word

1、python-docx 库介绍

* 该模块儿可以创建、修改 Word（.docx）文件；

* 此模块儿不属于 python 标准库，需要单独安装；

* python-docx 使用官网： https://python-docx.readthedocs.io/en/latest/；

* 我们在安装此模块儿使用的是 pip install python-docx，但是在导入的时候是 import docx；

2、Python 读取 Word 文档内容

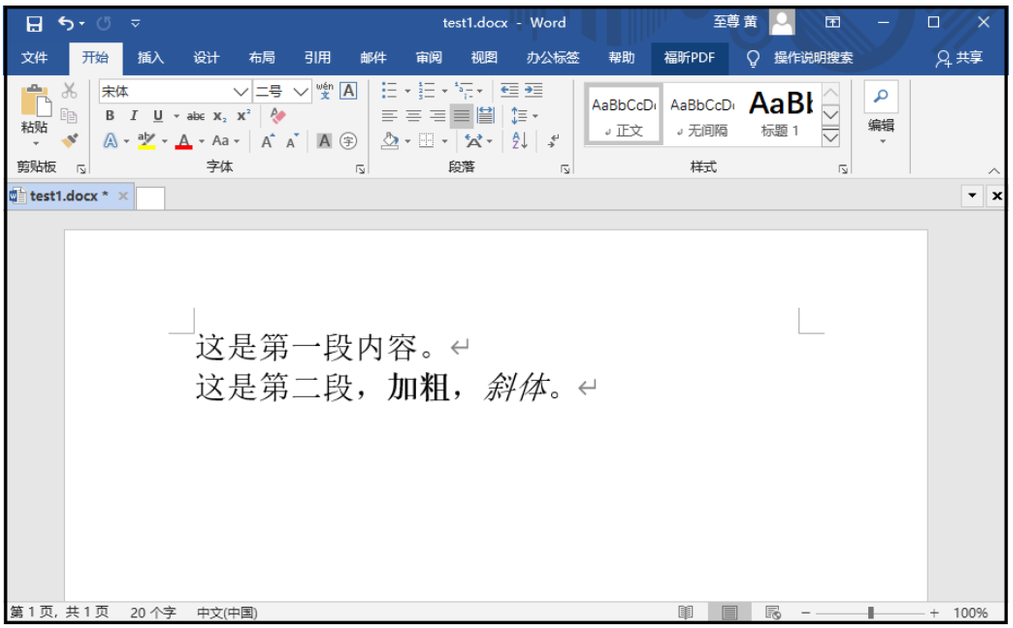
* 注意：每进行一个操作，必须保存一下，否则等于白做；

1）word 文档结构介绍

2）python-docx 提取文字和文字块儿

① python-docx 提取文字

有一个这样的 docx 文件，我们想要提取其中的文字，应该怎么做？



代码如下：

```
from docx import Document
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
print(doc.paragraphs)
for paragraph in doc.paragraphs:
    print(paragraph.text)
```

结果如下：

```
In [95]: from docx import Document

         doc = Document(r"G:\6Tipdm\7python办公自动化\concat_word\test1.docx")
         print(doc.paragraphs)

         for paragraph in doc.paragraphs:
             print(paragraph.text)

[<docx.text.paragraph.Paragraph object at 0x00000242CC2E4780>,
 <docx.text.paragraph.Paragraph object at 0x00000242CC2E4F98>]
这是第一段内容。
这是第二段，加粗，斜体。
```

② python-docx 提取文字块儿

```
from docx import Document
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
print(doc.paragraphs)
paragraph = doc.paragraphs[0]
runs = paragraph.runs
print(runs)
for run in paragraph.runs:
    print(run.text)
```

```
paragraph = doc.paragraphs[1]
runs = paragraph.runs
print(runs)
for run in paragraph.runs:
    print(run.text)
```

结果如下：

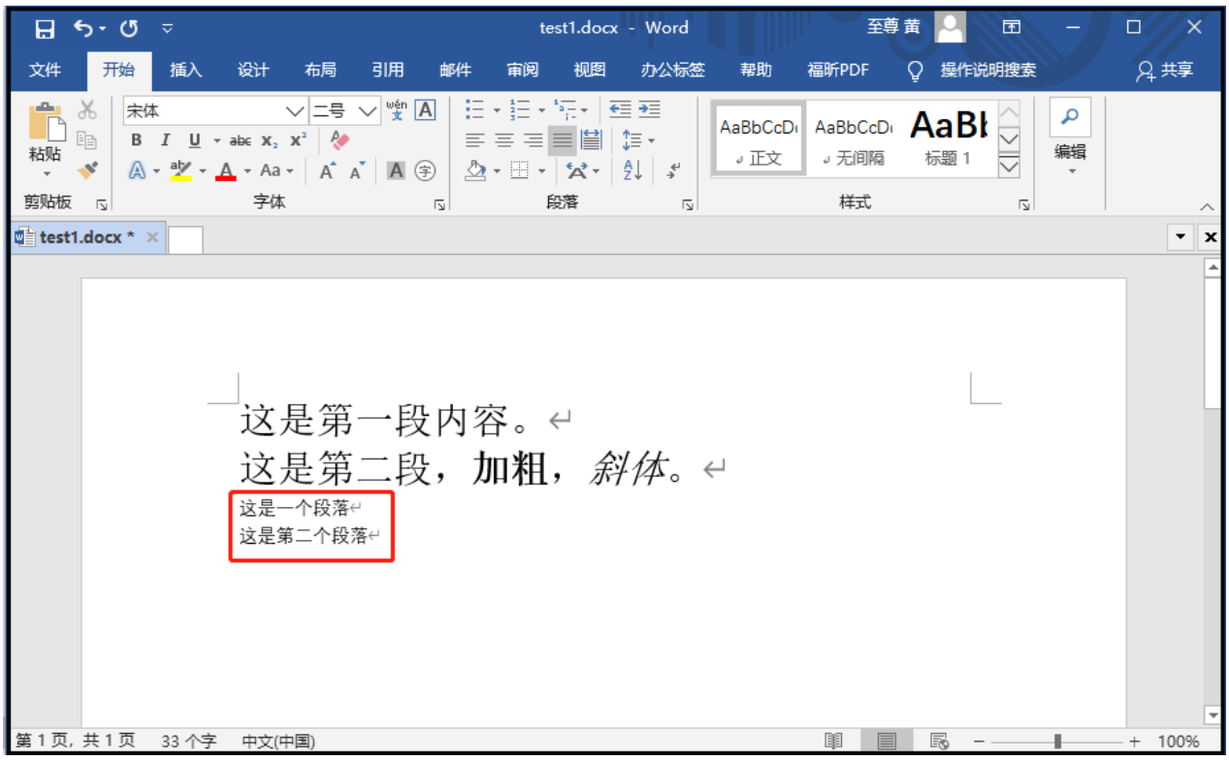
```
In [99]: from docx import Document

         doc = Document(r"G:\6Tipdm\7python办公自动化\concat_word\test1.docx")
         print(doc.paragraphs)
         paragraph = doc.paragraphs[0]
         runs = paragraph.runs
         print(runs)
         for run in paragraph.runs:
             print(run.text)

         [<docx.text.paragraph.Paragraph object at 0x00000242CC2E44A8>,
          <docx.text.paragraph.Paragraph object at 0x00000242CC2E4470>]
         [<docx.text.run.Run object at 0x00000242CC2E49E8>]
         这是第一段内容。

In [100]: paragraph = doc.paragraphs[1]
          runs = paragraph.runs
          print(runs)
          for run in paragraph.runs:
              print(run.text)

          [<docx.text.run.Run object at 0x00000242CC2E4780>, <docx.text.run.Run object at 0x00000242CC2E4748>,
           <docx.text.run.Run object at 0x000 00242CC2E4240>, <docx.text.run.Run object at 0x00000242CC2E4128>,
           <docx.text.run.Run object at 0x00000242CC2E4BA8>]
          这是第二段，
          加粗
          ，
          斜体
          。
```

3）利用 Python 向 Word 文档写入内容

① 添加段落

```
from docx import Document
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
# print(doc.add_heading("一级标题", level=1))  添加一级标题的时候出错，还没有解决！
paragraph1 = doc.add_paragraph("这是一个段落")
paragraph2 = doc.add_paragraph("这是第二个段落")
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
"""
添加段落的时候，赋值给一个变量，方便我们后面进行格式调整；
"""
```

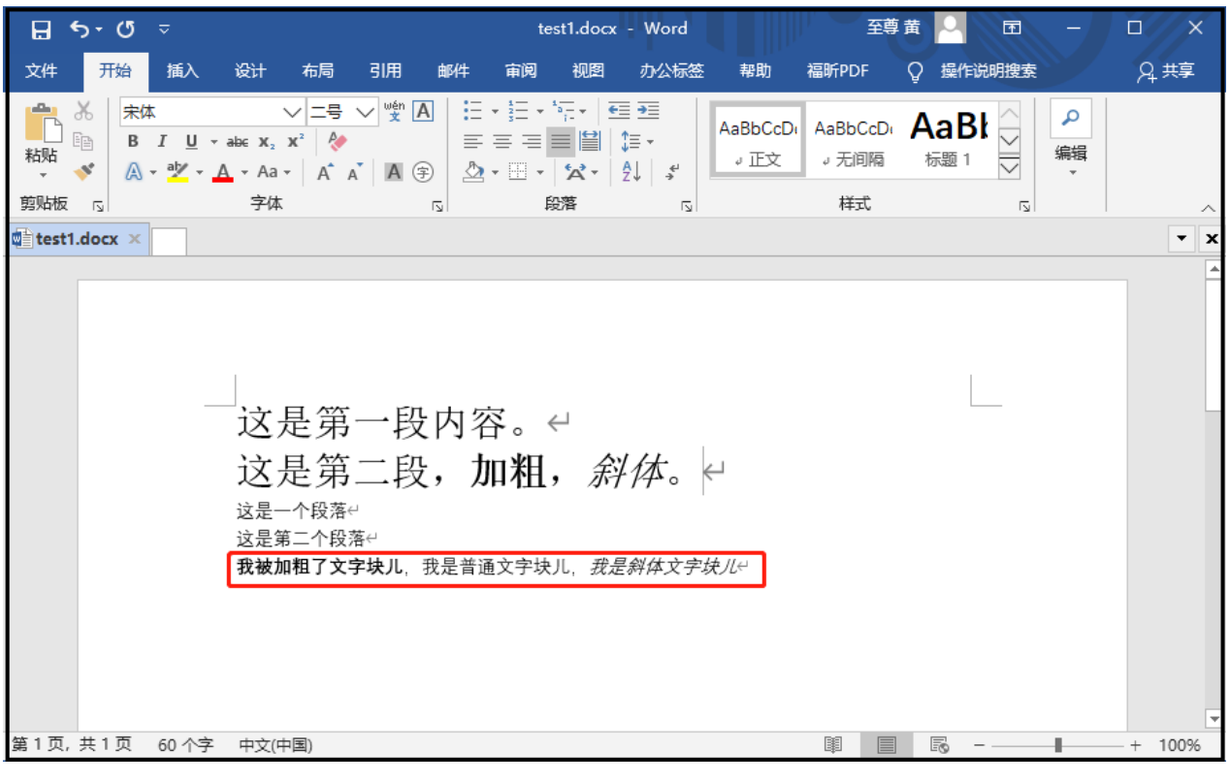
结果如下：



② 添加文字块儿

```
from docx import Document
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
# 这里相当于输入了一个空格，后面等待着文字输入
paragraph3 = doc.add_paragraph()
paragraph3.add_run("我被加粗了文字块儿").bold = True
paragraph3.add_run("，我是普通文字块儿，")
paragraph3.add_run("我是斜体文字块儿").italic = True
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
```

结果如下：

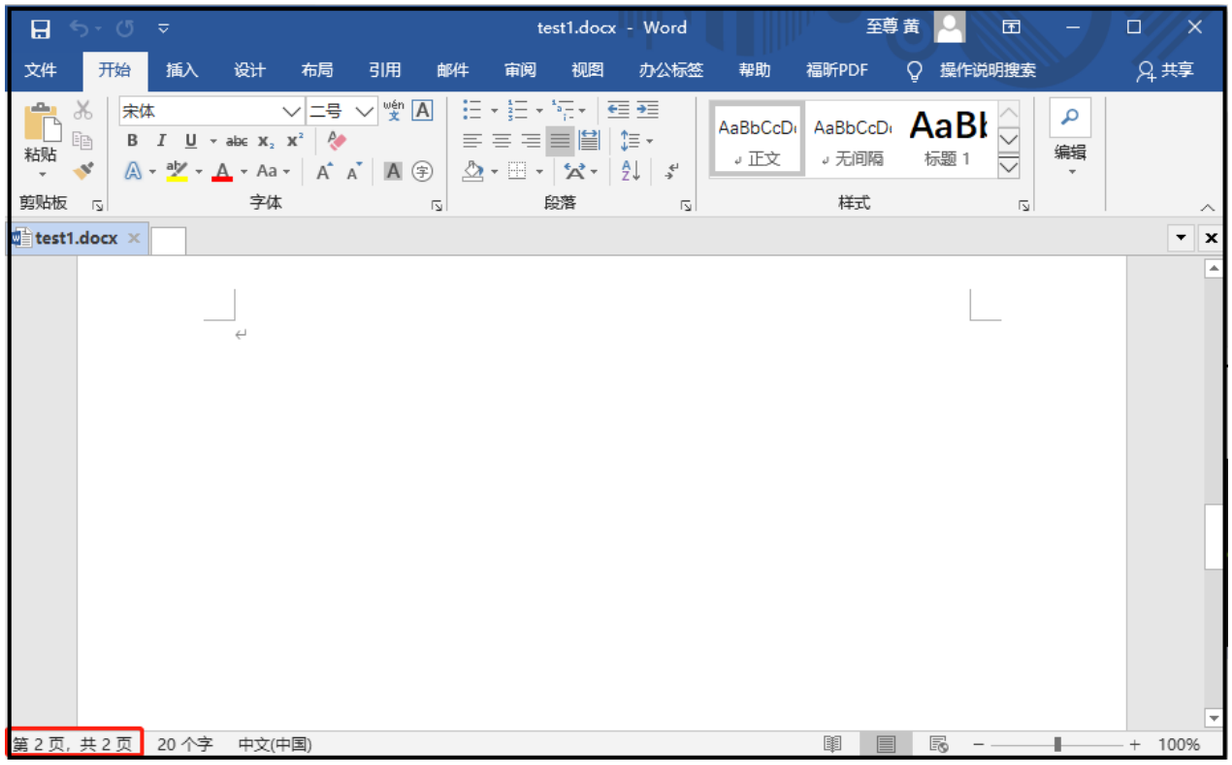


③ 添加一个分页

```
from docx import Document
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
doc.add_page_break()
```

```
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
```

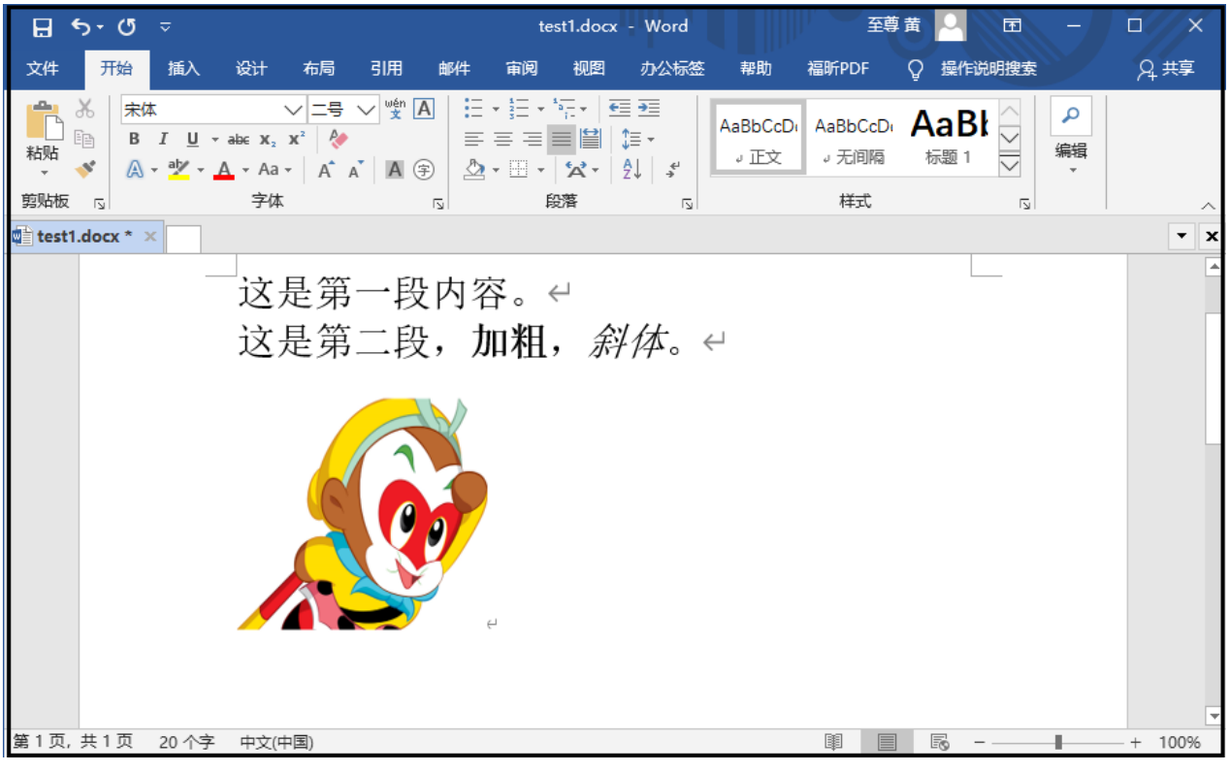
结果如下：



④ 添加图片

```
from docx import Document
from docx.shared import Cm
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
doc.add_picture(r"G:\6Tipdm\7python 办公自动化
\concat_word\sun_wu_kong.png",width=Cm(5),height=Cm(5))
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
"""
Cm 模块，用于设定图片尺寸大小
"""
```

结果如下：

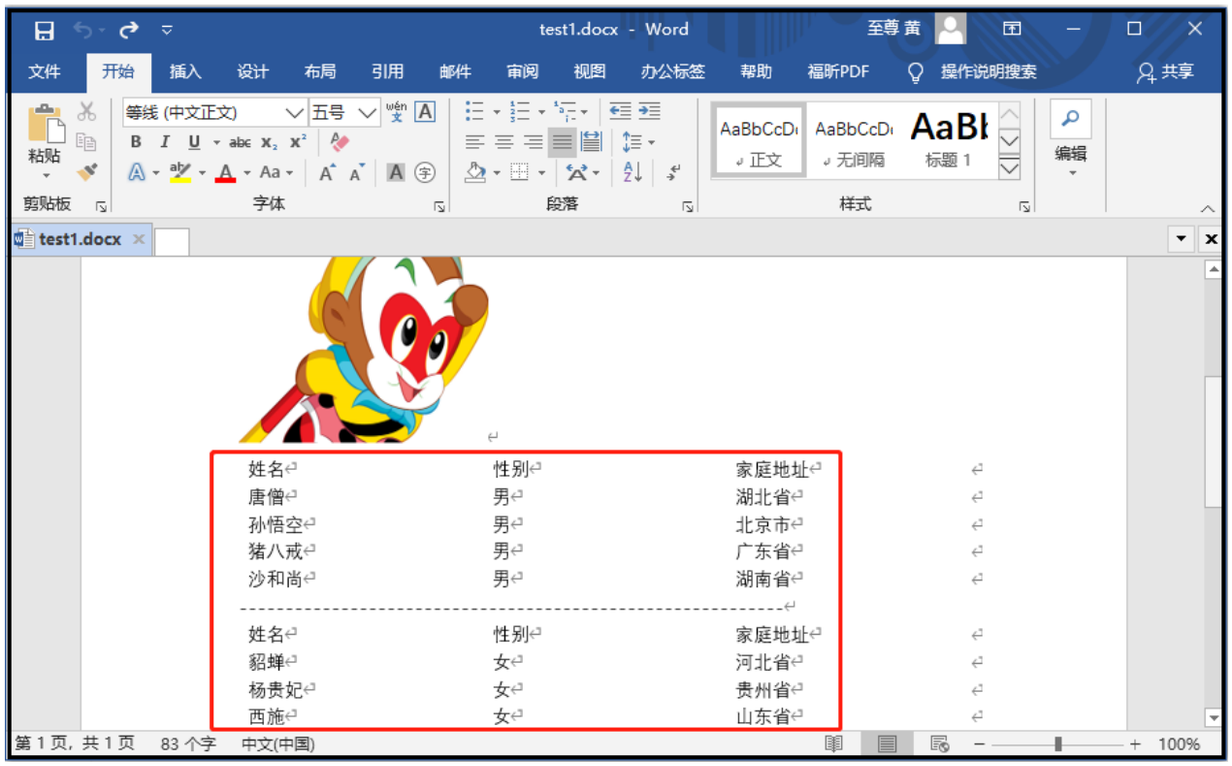


⑤ 添加表格

```
from docx import Document
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
list1 = [
        ["姓名","性别","家庭地址"],
        ["唐僧","男","湖北省"],
        ["孙悟空","男","北京市"],
        ["猪八戒","男","广东省"],
        ["沙和尚","男","湖南省"]
       ]
list2 = [
        ["姓名","性别","家庭地址"],
        ["貂蝉","女","河北省"],
        ["杨贵妃","女","贵州省"],
        ["西施","女","山东省"]
```

```
        ]
    table1 = doc.add_table(rows=5,cols=3)
    for row in range(5):
        cells = table1.rows[row].cells
        for col in range(3):
            cells[col].text = str(list1[row][col])
doc.add_paragraph("----------------------------------------------------------") table2 =
doc.add_table(rows=4,cols=3)
for row in range(4):
    cells = table2.rows[row].cells
        for col in range(3):
            cells[col].text = str(list2[row][col])
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
```

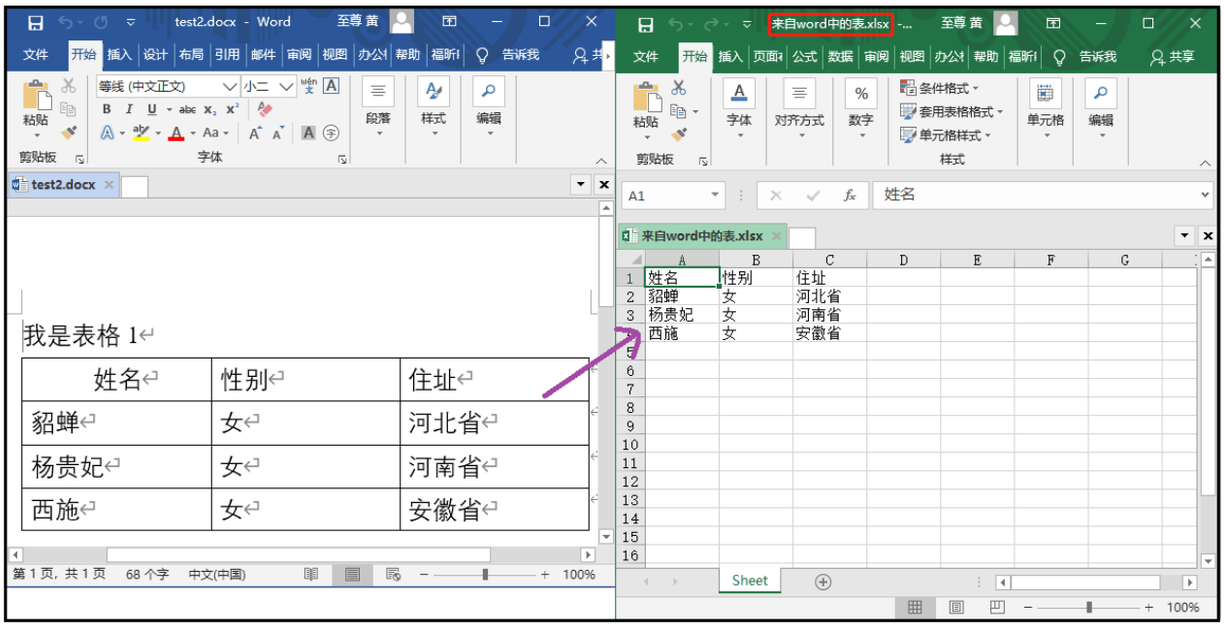
结果如下：



⑥ 提取 word 表格，并保存在 excel 中(很重要)

```
from docx import Document
from openpyxl import Workbook
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test2.docx")
t0 = doc.tables[0]
workbook = Workbook()
sheet = workbook.active
for i in range(len(t0.rows)):
    list1 = []
    for j in range(len(t0.columns)):
        list1.append(t0.cell(i,j).text)
        sheet.append(list1)
workbook.save(filename = r"G:\6Tipdm\7python 办公自动化\concat_word\来自 word 中的表.xlsx")
```

结果如下：

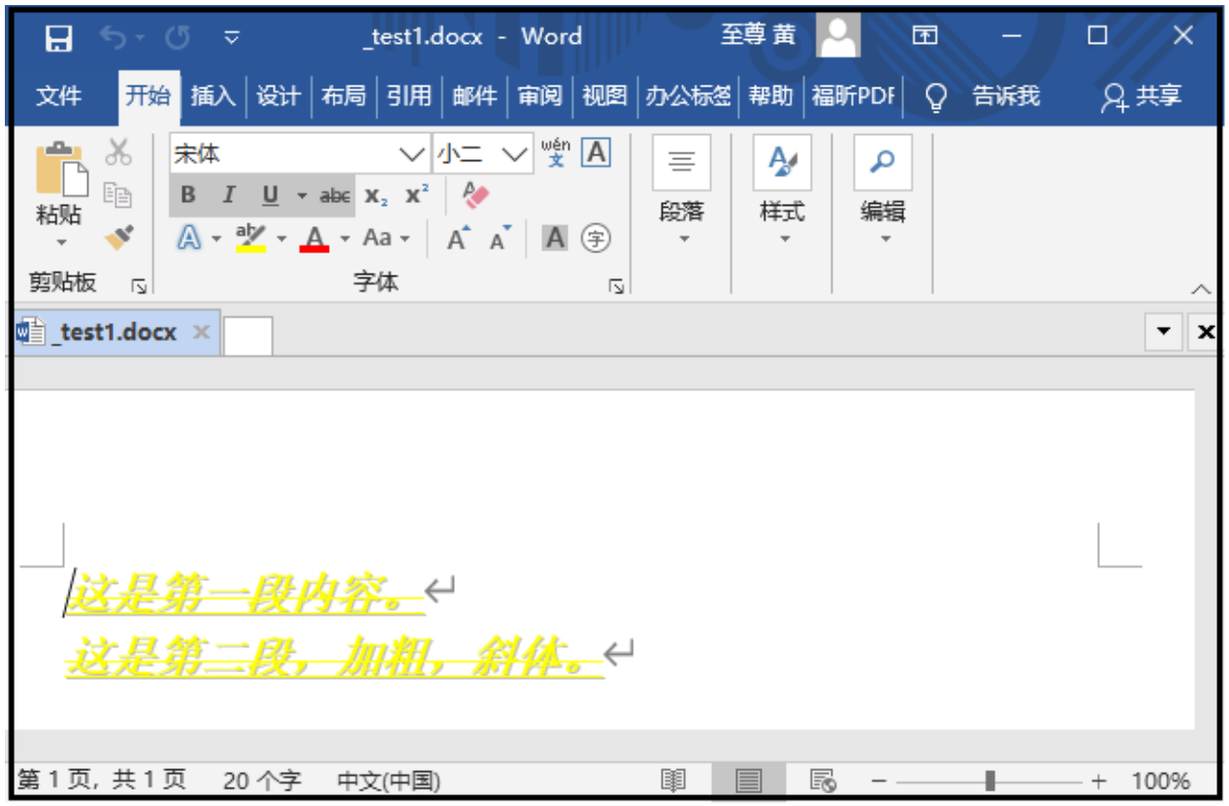


3、利用 Python 调整 Word 文档样式

1）修改文字字体样式

```
from docx import Document
from docx.shared import Pt,RGBColor
from docx.oxml.ns import qn
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test2.docx")
for paragraph in doc.paragraphs:
    for run in paragraph.runs:
        run.font.bold = True
        run.font.italic = True
        run.font.underline = True
        run.font.strike = True
        run.font.shadow = True
        run.font.size = Pt(18)
        run.font.color.rgb = RGBColor(255,255,0)
        run.font.name = "宋体"
        # 设置像宋体这样的中文字体，必须添加下面 2 行代码
        r = run._element.rPr.rFonts
        r.set(qn("w:eastAsia"),"宋体")
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\_test1.docx")
```

结果如下：

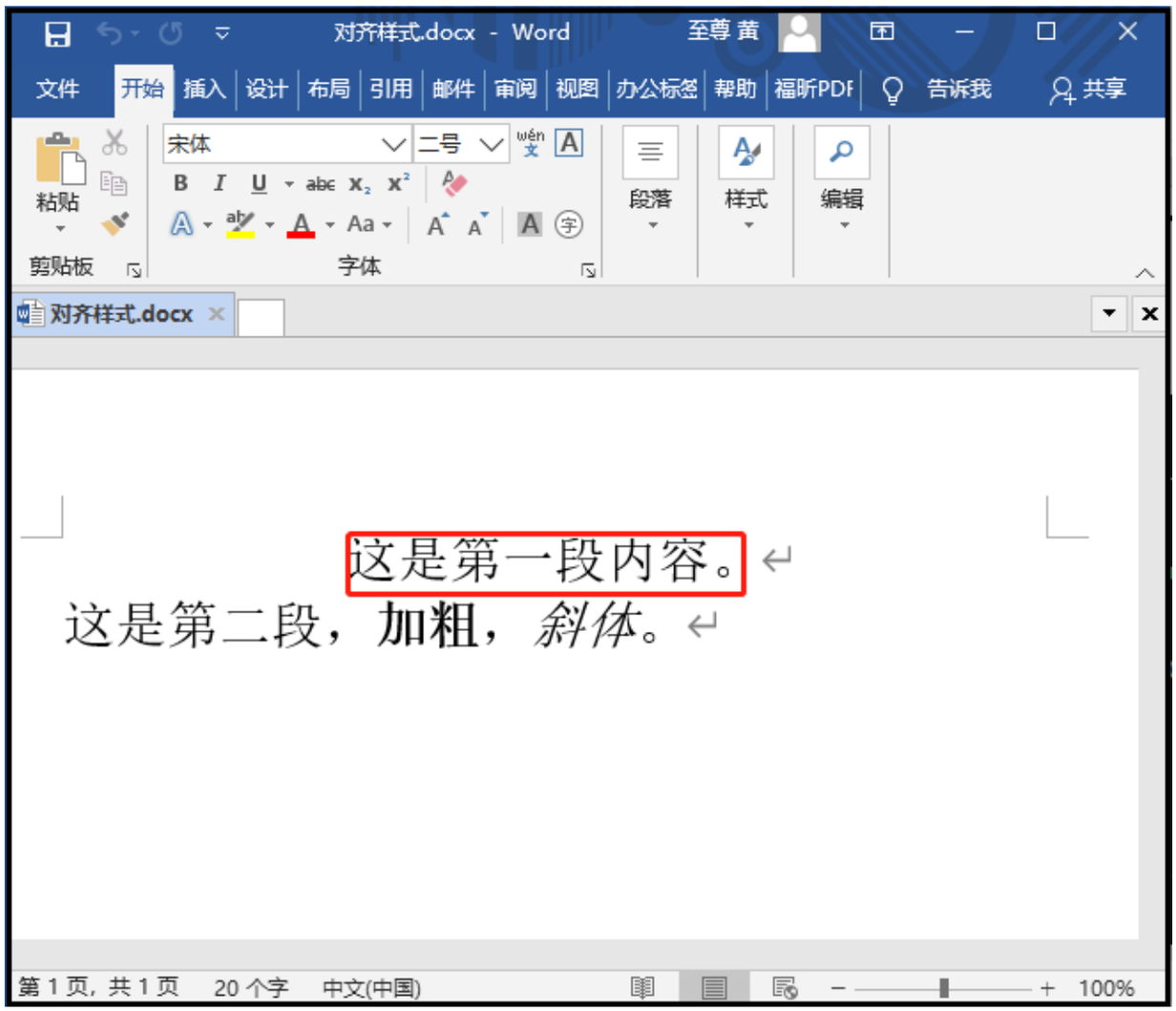


2）修改段落样式

① 对齐样式

```
from docx import Document
from docx.enum.text import WD_ALIGN_PARAGRAPH
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
print(doc.paragraphs[0].text)
doc.paragraphs[0].alignment = WD_ALIGN_PARAGRAPH.CENTER
# 这里设置的是居中对齐
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\对齐样式.docx")
"""
居中对齐是其中一种样式，这里还有其他选择，自己百度了解：
LEFT,CENTER,RIGHT,JUSTIFY,DISTRIBUTE,JUSTIFY_MED,JUSTIFY_HI,JUSTIFY_L
OW,THAI_JUSTIFY
"""
```

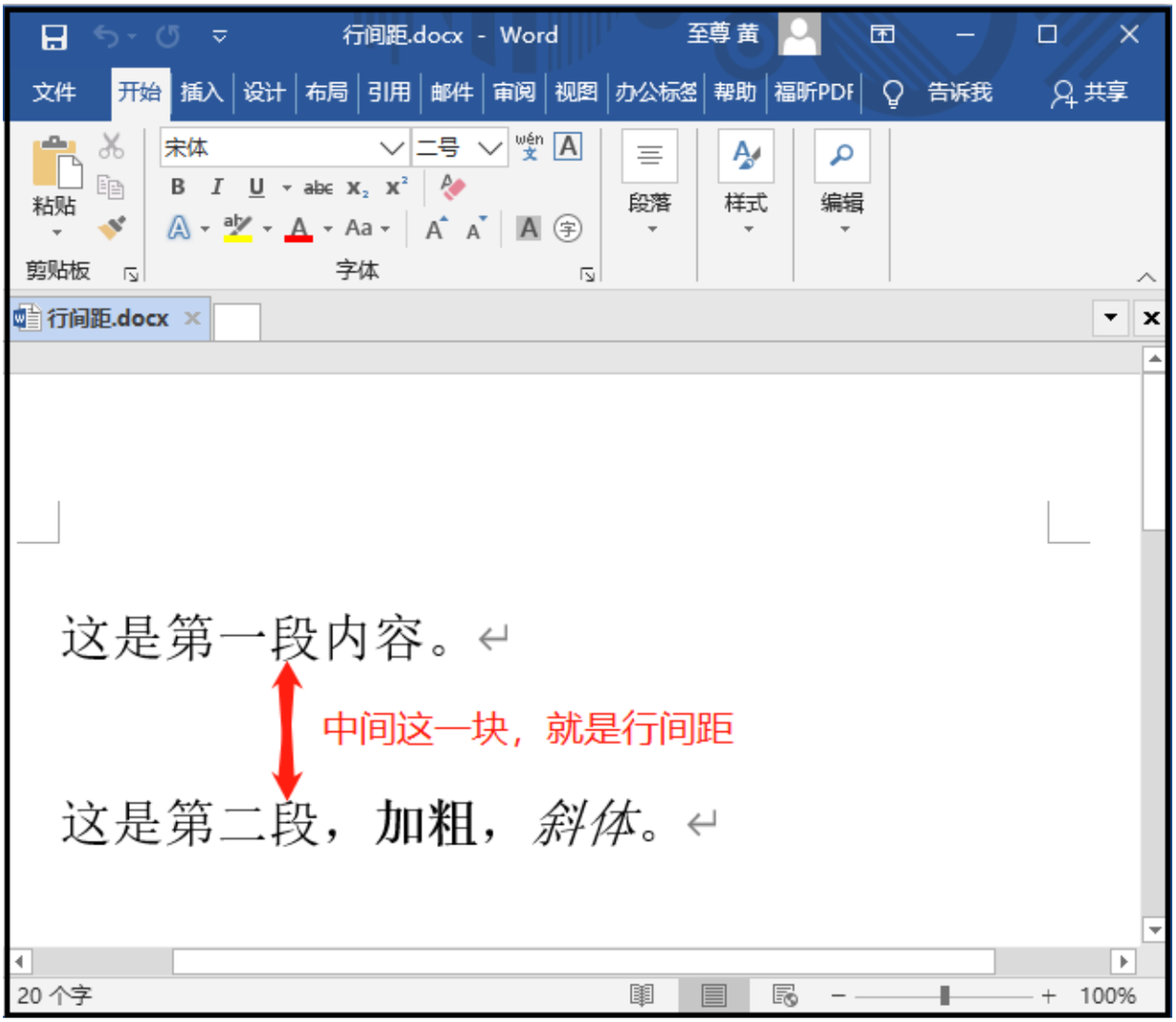
结果如下：



② 行间距调整

```
from docx import Document
from docx.enum.text import WD_ALIGN_PARAGRAPH
doc = Document(r"G:\6Tipdm\7python 办公自动化\concat_word\test1.docx")
for paragraph in doc.paragraphs:
    paragraph.paragraph_format.line_spacing = 5.0
doc.save(r"G:\6Tipdm\7python 办公自动化\concat_word\行间距.docx")
```

结果如下：



③ 段前与段后间距

* 这里提供代码，自行下去检验

段前与段后间距

paragraph.paragraph_format.space_before = Pt(12)

Pt(12)表示12磅

```
paragraph.paragraph_format.space_before = Pt(12)
paragraph.paragraph_format.space_after = Pt(12)
```