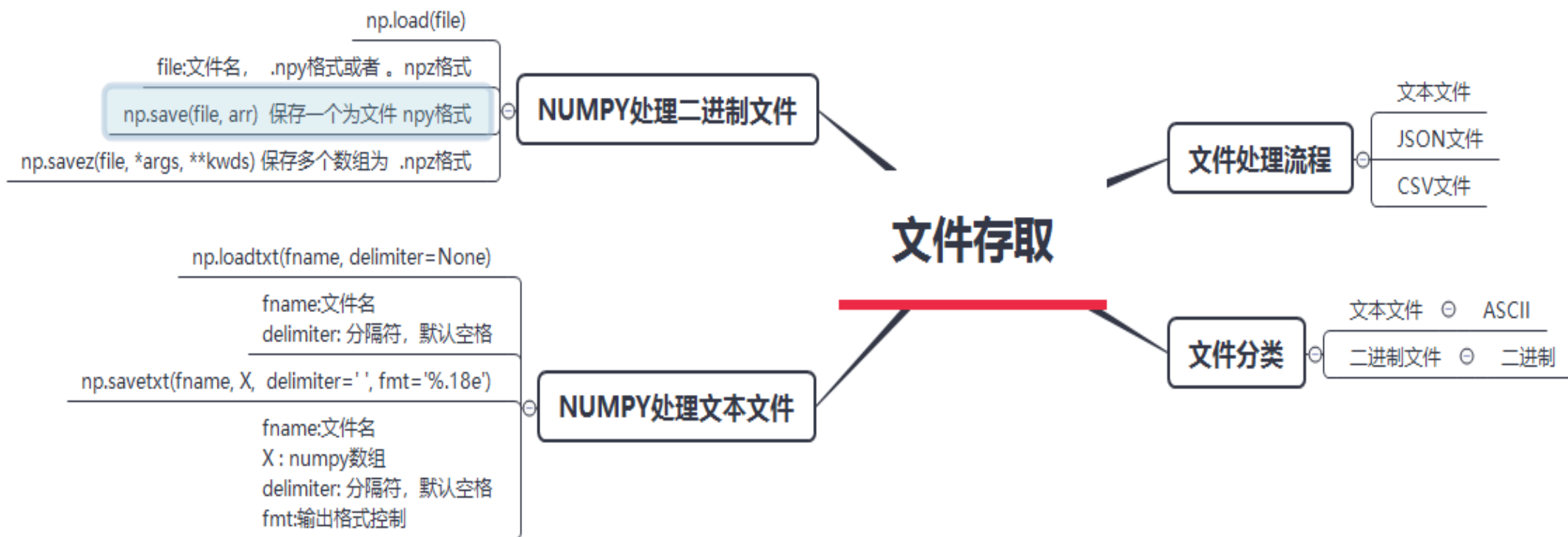


Python科学计算库Numpy之

10-文件读写

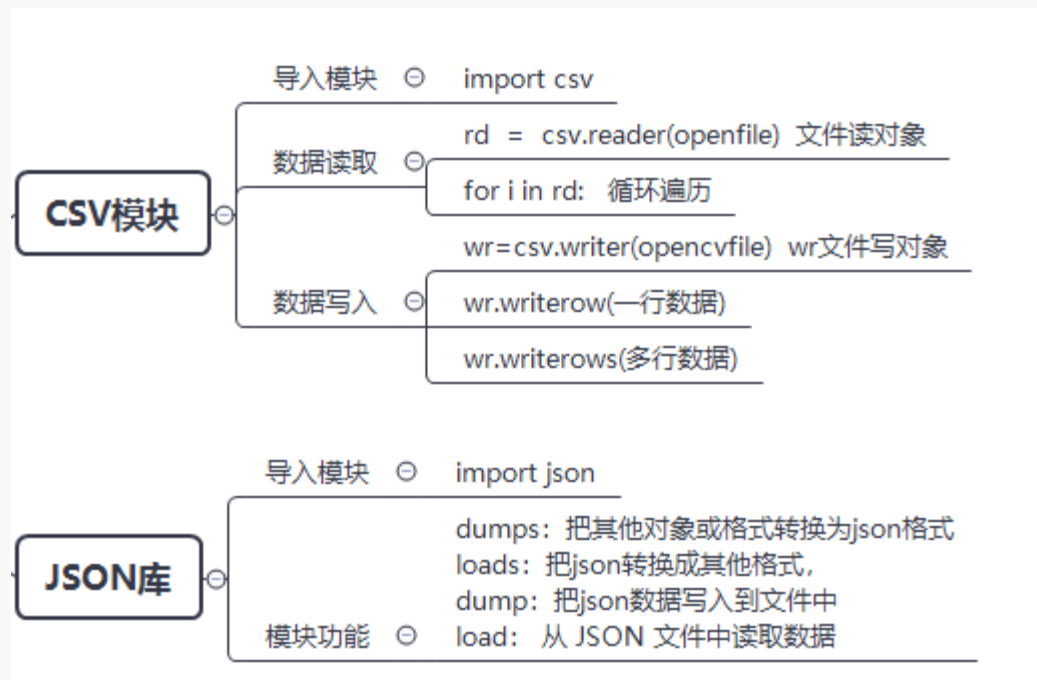


知识结构图



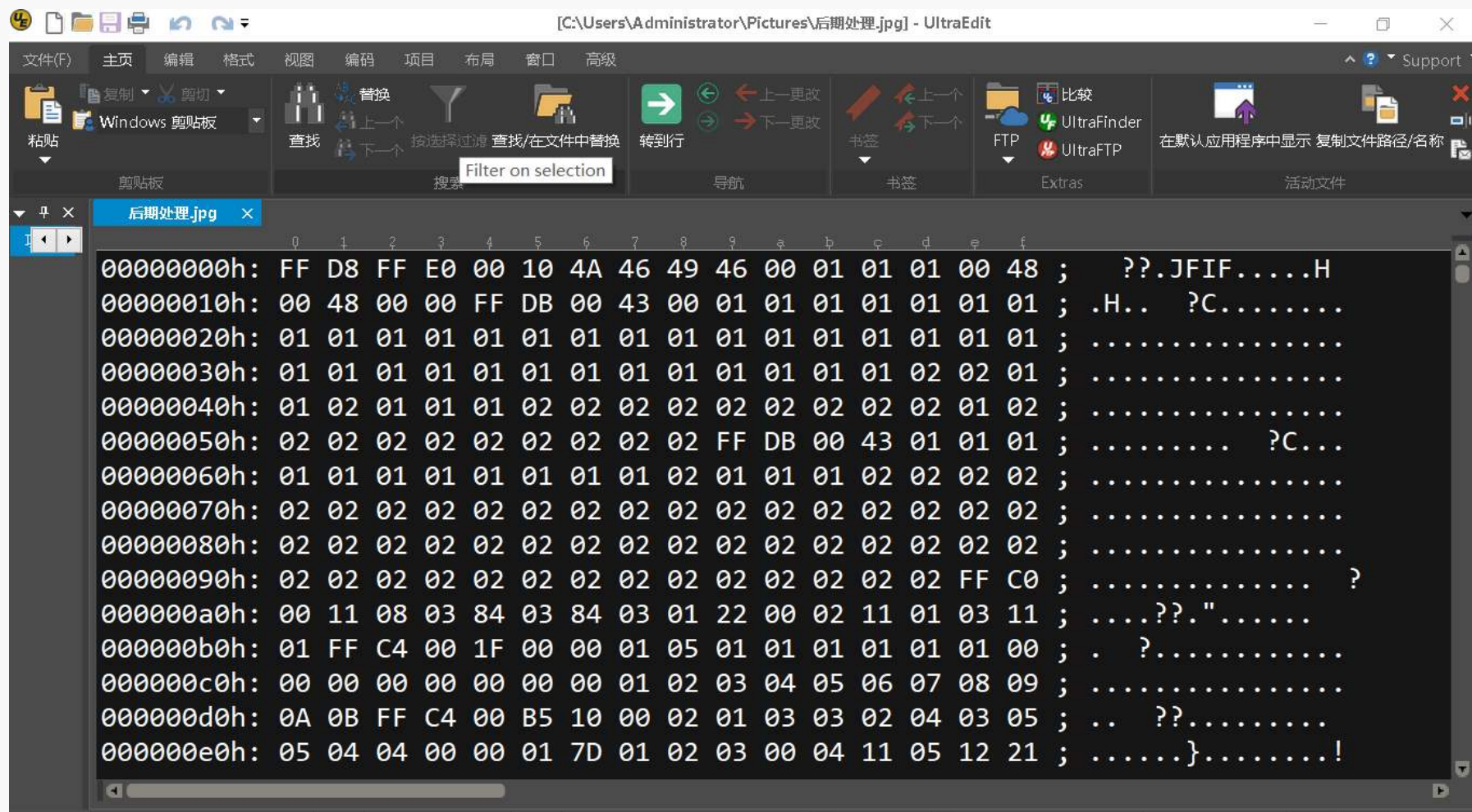
截图(Alt + A)

文件复习



以文本格式存储数值

二进制文件



The screenshot shows the UltraEdit text editor with a file named "后期处理.jpg" (Post-processing.jpg) open. The editor displays a hex dump of the file's content. The hex dump shows the JFIF header and the start of the image data. The first few lines of the hex dump are:

```
00000000h: FF D8 FF E0 00 10 4A 46 49 46 00 01 01 01 00 48 ; ?? .JFIF....H
00000010h: 00 48 00 00 FF DB 00 43 00 01 01 01 01 01 01 01 ; .H.. ?C.....
00000020h: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 ; .....
00000030h: 01 01 01 01 01 01 01 01 01 01 01 01 02 02 01 01 ; .....
00000040h: 01 02 01 01 01 02 02 02 02 02 02 02 02 02 01 02 ; .....
00000050h: 02 02 02 02 02 02 02 02 02 FF DB 00 43 01 01 01 ; ..... ?C...
00000060h: 01 01 01 01 01 01 01 01 02 01 01 01 02 02 02 02 ; .....
00000070h: 02 02 02 02 02 02 02 02 02 02 02 02 02 02 02 02 ; .....
00000080h: 02 02 02 02 02 02 02 02 02 02 02 02 02 02 02 02 ; .....
00000090h: 02 02 02 02 02 02 02 02 02 02 02 02 02 FF C0 ; ..... ?
000000a0h: 00 11 08 03 84 03 84 03 01 22 00 02 11 01 03 11 ; ....??. ".....
000000b0h: 01 FF C4 00 1F 00 00 01 05 01 01 01 01 01 01 00 ; . ?.....
000000c0h: 00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09 ; .....
000000d0h: 0A 0B FF C4 00 B5 10 00 02 01 03 03 02 04 03 05 ; .. ??.....
000000e0h: 05 04 04 00 00 01 7D 01 02 03 00 04 11 05 12 21 ; .....}.....!
```

文本文件与二进制文件

文本文件与二进制文件对比

对比short型数据1268在两种不同文件类型的存储形式

short型数	二进制文件	字符文件
1268	2 字节	4 字节

00110001	00110010	00110110	00111000
'1' 的ASCII码	'2' 的ASCII码	'6' 的ASCII码	'8' 的ASCII码

每个字符通过相应的编码(ASCII)存储在文件中

00000100	11110100
1268的二进制数	

直接把内存数据以二进制形式保存

`np.loadtxt(fname, delimiter=None)`

功能:

从文本文件加载数据

返回值:

一个numpy.Ndarray数组

参数:

`fname`: 文件名

`delimiter`: 分隔符, 默认空格

`np.savetxt(fname, X, delimiter=' ', fmt='%.18e')`

功能:

将X数组写入文本文件`fname`

`delimiter`: 分隔符, 默认空格

`fmt`: 输出格式控制

```
%%writefile myfile.txt
2.1, 2.3, 3.2, 1.3, 3.1
6.1, 3.1, 4.2, 2.3, 1.8
```

```
data = np.loadtxt('myfile.txt', delimiter=',')
data
```

```
array([[2.1, 2.3, 3.2, 1.3, 3.1],
       [6.1, 3.1, 4.2, 2.3, 1.8]])
```

```
data = np.array([[1, 2],
                 [3, 4]])
```

```
np.savetxt('out.txt', data, fmt="%d") #保存为整数
```

```
# linux
#!cat out.txt
```

```
# windows
!type out.txt
```

```
1 2
3 4
```


Numpy 二进制文件格式

单个的数组保存为 .npy 格式

多个数组保存为 .npz 格式

与文本格式不同，二进制格式保存了数组的 shape, dtype 信息，以便完全重构出保存的数组。

保存的方法：

`np.save(file, arr)` 保存单个数组，.npy 格式

`np.savez(file, *args, **kwargs)` 保存多个数组，无压缩的 .npz 格式

`np.savez_compressed(file, *args, **kwargs)` 保存多个数组，有压缩的 .npz 格式

读取的方法：

`np.load(file)`

对于 .npy，返回保存的数组，对于 .npz，

返回一个名称-数组对组成的字典。

Numpy二进制文件

```
a = np.array([[1.0, 2.0], [3.0, 4.0]])
```

```
fname = 'afile.npy'  
np.save(fname, a)
```

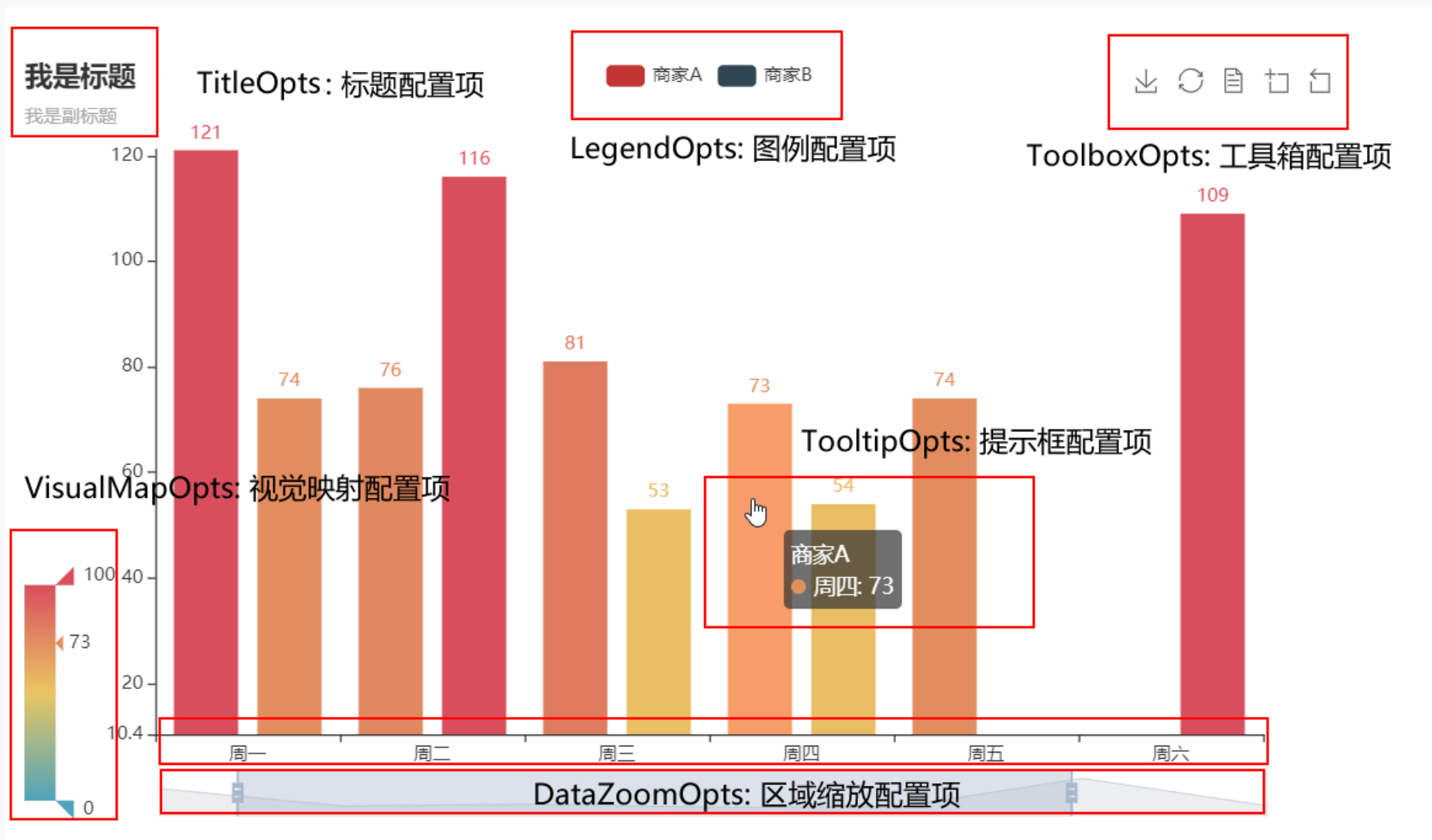
```
! cat afile.npy
```

```
NPY 1.0 {'descr': '<f8', 'fortran_order': False, 'shape': (2, 2), }  
??@?@
```

```
aa = np.load(fname)  
aa
```

```
array([[1., 2.],  
       [3., 4.]])
```


文件复习



谢谢!

