选13、正则表达式

1、杭州区域号牌识别

2021年3月1日起，杭州推出区域号牌，如何识别杭州普通号牌和区域号牌呢?
“浙A区域号牌”的组合方式现阶段为“浙A”后的第2、5位或第3、4位均为字母，其余为数字，如：“浙A·5B55B”或“浙A·55BB5”。至于说杭州区域牌照和正常牌照有什么区别，最大的区别是限行规定有区别。

现在列表brand=[’浙A2B45B’,’浙A23EB2’,’沪A23EB2’,’浙B3212F’,’浙A3U2U8’]
要从中筛选出杭州区域号牌，该如何编程实现呢?

2、方法1：传统方法实现

```
brand=['浙A2B45B','浙A23EB2','沪A23EB2','浙B3212F','浙A3U2U8']
for b in brand:
    if b[:2]=='浙A' and '0'<=b[2]<='9' and 'A'<=b[3]<='Z' and '0'<=b[4]<='9' and '0'<=b[5]<='9' and 'A'<=b[6]<='Z' or\
        b[:2]=='浙A' and '0'<=b[2]<='9' and '0'<=b[3]<='9' and 'A'<=b[4]<='Z' and 'A'<=b[5]<='Z' and '0'<=b[6]<='9':
            print(b)
```

```
浙A2B45B
浙A23EB2
```

看到条件写的够复杂吧

3、正则表达式方法

文本匹配的时候，有许多规则，这些规则如果直接用逻辑运算连接，会异常复杂。
因此有人发明了正则表达式，通过正则表达式来匹配文本格式会容易很多。
导入模块：
import re
构建正则表达式：
正则表达式是以r进行修饰的一段字符串:
r1=r’\d\D’ #匹配一个数字和一个非数字
通过re.match得到匹配结果,如果匹配不到则返回None

常用正则表达式语法：

语法	说明	表达式实例	完整匹配的字符串
字符			
一般字符	匹配自身	abc	abc
.	匹配任意除换行符"\n"外的字符。 在DOTALL模式中也能匹配换行符。	a.c	abc
\	转义字符，使后一个字符改变原来的意思。 如果字符串中有字符*需要匹配，可以使用*或者字符集[*]。	a\.c a\\c	a.c a\c
[...]	字符集（字符类）。对应的位置可以是字符集中任意字符。 字符集中的字符可以逐个列出，也可以给出范围，如[abc]或[a-c]。第一个字符如果是^则表示取反，如[^abc]表示不是abc的其他字符。 所有的特殊字符在字符集中都失去其原有的特殊含义。在字符集中如果要使用]、-或^，可以在前面加上反斜杠，或把]、-放在第一个字符，把^放在非第一个字符。	a[bcd]e	abe ace ade
预定义字符集（可以写在字符集[...]中）			
\d	数字：[0-9]	a\dc	a1c
\D	非数字：[^\d]	a\Dc	abc
\s	空白字符：[<空格>\t\r\n\f\v]	a\sc	a c
\S	非空白字符：[^\s]	a\Sc	abc
\w	单词字符：[A-Za-z0-9_]	a\wc	abc
\W	非单词字符：[^\w]	a\Wc	a c
数量词（用在字符或(...)之后）			
*	匹配前一个字符0或无限次。	abc*	ab abccc
+	匹配前一个字符1次或无限次。	abc+	abc abccc
?	匹配前一个字符0次或1次。	abc?	ab abc
{m}	匹配前一个字符m次。	ab{2}c	abbc
{m,n}	匹配前一个字符m至n次。 m和n可以省略：若省略m，则匹配0至n次；若省略n，则匹配m至无限次	ab{1,2}c	abc abbc

4、通过正则表达式筛选杭州区域号牌

```
#杭州区域号牌识别
import re
brand=['浙A2B45B','浙A23EB2','沪A23EB2','浙B3212F','浙A3U2U8']
r1=r"浙A\d\D\d\d\D"
r2=r"浙A\d\d\D\D\d"
for b in brand:
    if re.match(r1,b) or re.match(r2,b):
        print(b)
```