"""
正则表达式
1、基础匹配
学习目标：
    1、了解什么是正则表达式
    2、掌握re模块的基础使用
正则表达式，又称为规则表达式Regular Expression，是使用单个字符串来描述、匹配某个句法规则的字符串，常用来检索、替换那些复合某个模式（规则）的文本。

简单来说，正则表达式就是使用：字符串定义归职责，并通过规则去验证字符串是否匹配。
如：验证一个字符串是否复合条件的电子邮箱地址，只需要配置号正则规则，即可匹配任意邮箱地址。
2536920213@qq.com
正则三个基础方法：
python正则表达式，使用re模块，并基于re模块中三个基础方法来做正则匹配。
(1)match
语法：
    re.match(匹配规则,被匹配的字符串)
    从被匹配字符串开头进行匹配，匹配成功返回匹配对象（包含匹配的信息），匹配不成功返回空。
(2)search
语法：
    re.search(匹配规则,被匹配的字符串)
    搜索整个字符串，找出匹配的。从前向后，找到第一个后，就停止，不会继续向后。如果整个字符串都找不到，

(3)findall
语法：
    re.findall(匹配规则,被匹配的字符串)
    匹配整个字符串，找出全部匹配项，如果找不到则返回空list:[]
"""
# 导入re模块
import re
# **********match**********
# s = "Python lanzhi Python shujia Python Python"
# result = re.match("Python",s)
# print(result) # <re.Match object; span=(0, 6), match='Python'>
# print(result.span()) # (0, 6)
# print(result.group()) # Python
# 注意：从第一个字符开始匹配，匹配成功返回匹配对象，匹配不成功返回None

**********search**********
s = "11111Python lanzhi Python shujia Python Python"
result = re.search("Python",s)

```
# print(result)  # <re.Match object; span=(1, 7), match='Python'>
# print(result.span())  # (5, 11)
# print(result.group())  # Python


# ************************findall*****************************
s = "11111Python lanzhi Python shujia Python Python"
result = re.findall("java",s)
print(result)  # ['Python', 'Python', 'Python', 'Python']

# "A", "I", "L", "M", "S", "X", "U"
# re.A
# re.I
# re.L
# re.M
# re.S
# re.U
# re.X



"""
https://www.runoob.com/regexp/regexp-metachar.html
正则表达式：
2、元字符匹配
学习目标：
    1、掌握正则表达式的各类元字符规则
    2、了解字符串的r标记的作用
单字符匹配：
. 匹配任意1个字符（除了\n），\. 匹配点的本身
[] 匹配[]中列举的字符
\d 匹配数字，即0-9
\D 匹配非数字
\s 匹配空白，即空格、tab键
\w 匹配单词字符即，a-z A-Z 0-9 _
\W 匹配非字符

元字符匹配
* 匹配前一个规则的字符出现0至无数次
+ 匹配前一个规则的字符出现1至无数次
? 匹配前一个规则的字符出现0次或1次
{n} 精确匹配n次
{n,} 匹配前一个规则的字符出现最少n次
```

{n,m} 匹配前一个规则的字符出现n到m次

边界匹配
^ (脱字符) 匹配字符串开头
$ (美元符) 匹配字符串的结尾
\b 匹配一个单词的边界
\B 匹配非单词边界

分组匹配
| 表示或的关系，匹配左右任意一个表达式
() 定义子表达式或捕获组。将括号中的字符作为一个分组

```
"""
import re
s = "lanz22hi$*shujia1@@python  @@#!!666##shuji33a"
result = re.findall(r"\d",s)  # r标记，表示字+符串中转义字符无效，就普通字符的意思
print(result)
print(type(result))


# 找出特殊字符
result2 = re.findall(r'\W',s)
print(result2)


# **********案例**********
# 匹配手机号码
# 手机号都是11位，以1开头，第二位可以是3-9，后面9位可以是0-9
text1 = "我的手机号是：23800000000机号都 17398383071 机号都13800000002 机号都13800000003机号都
pattern = r'1[3-9]\d{9}'
"""
1 :手机号的第一位是固定的为1
[3-9]  :第二位数字是3到9之间的任意一个数字
\d{9} 后面跟着9位任意数字，\d表示数字，{9}表示前面的规则重复9次
"""
phone_number = re.findall(pattern,text1)
print(phone_number)


# 匹配任意网址
# https://www. baidu.com/
# https://www.runoob.com/regexp/regexp-metachar.html

urls = "https://www.baidu.com/s?ie=utf-8&f=8&rsv_bp=1&rsv_idx=1&tn=baidu&wd=python&fenlei
```

```
"""
1、http[s]?   -------> http 或者https
http:普通的字符序列，直接匹配字符串http
[s]? 匹配s字符出现0次或1次
2、://
这是普通字符序列，直接匹配字符串://   (它是url中协议于域名之间的分隔符)
# 使用捕获组的方式
3、(?: .....)
(?:) 是一个非捕获分组，它将括号内的内容作为一个整体来处理，但不会保存匹配搭配的内容。这个分组包含了多
url包含域名、路径、查询参数等部分
4、[a-zA-Z]
这是一个字符类，匹配任意的大小写英文字母，在url中，域名、路径等部分可能包含字母
5、[0-9]
同样是字符类，匹配任意数字，url中端口号、参数值等部分可能会出现数字
6、[$-_@.&+]
字符类，匹配这些特定的符号 $、-、_、@、.、&、+。这些符号在url中较为常见，如&常用分隔查询参数 .用于分
7、[!*\\(\\),]
字符类，匹配符号!、*、（、）、,.在url中，这些符号也可能作为合法的字符出现,注意的是(和)是正则表达式中
8、(?:%[0-9a-fA-F][0-9a-fA-F])
这是一个非捕获组，用于匹配url中的百分号编码（URL）编码
% 匹配的是 %,它就是百分号编码的起始符号
[0-9a-fA-F] 字符类，匹配十六进制数字（0-9、a-f、A-F）。百分号编码后会跟着两个十六进制的数字。
9、+
量词

http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\\(\\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+
"""
pattern = r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\\(\\),]|(?:%[0-9a-fA-F][0-9a-fA-F]
result = re.findall(pattern,urls)
print(result)



# 匹配邮箱
# 只允许匹配qq、163、gmail这三种邮箱格式
# 2536920213@qq.com
# chen666@163.com
# chen666@gmail.com

text = "我的2536920213@qq.com邮箱地址是：test123@163.com,final@163.com和python@example.com a
# 2536920213  @  qq.com
"""
1、[a-zA-Z0-9._%+-]+
[]:这是字符类，用于定义一个字符集合，代表在这个集合里任选一个字符
```

a-zA-Z ： 表示匹配任意大小写英文字母

0-9 ： 表示匹配的任意数字

._%+- ： 表示匹配 . _ % + - 这些字符

+ ： 量词，表示匹配1次或多次

2、@

它是一个普通字符，在正则表达式里匹配邮箱地址的@符号。

3、(?:qq\.com|163\.com|gmail\.com)

(?:) 非捕获分组，它的作用是把多个元素组合成一个整体，不过不捕获匹配到的内容

| 这就是或运算符，两边的任意一个表达式

qq\.com、163\.com、gmail\.com 分别表示的邮箱域名

其中.在正则表达式里有特殊含义的，代表匹配换行符之外的任意字符。

所以使用\.来匹配 .

"""

```
pattern = r'[a-zA-Z0-9._%+-]+@(?:qq\.com|163\.com|gmail\.com)'
emails = re.findall(pattern,text)
print(emails)
```