https://segmentfault.com/a/1190000000725514

正则练习

1, 判断字符串是否是这样组成的，第一个必须是字母，后面可以是字母、数字、下划线

```
var reg = /^[a-zA-Z][a-zA-Z_0-9]{4,9}$/;
var str = "abd_234__fDFS";
alert(reg.test(str));
```

匹配中文字符的正则表达式： `[\u4e00-\u9fa5]`
评注：匹配中文还真是个头疼的事，有了这个表达式就好办了

匹配双字节字符(包括汉字在内)： `[^\x00-\xff]`
评注：可以用来计算字符串的长度（一个双字节字符长度计2，ASCII字符计1）

匹配空白行的正则表达式： `\n\s*\r`
评注：可以用来删除空白行

匹配HTML标记的正则表达式： `<(\S*?)[^>]*>.*?</\1>|<.*? />`
评注：网上流传的版本太糟糕，上面这个也仅仅能匹配部分，对于复杂的嵌套标记依旧无能为力

匹配首尾空白字符的正则表达式： `^\s*|\s*$`
评注：可以用来删除行首行尾的空白字符(包括空格、制表符、换页符等等)，非常有用的表达式
匹配Email地址的正则表达式： `\w+([-+.]\w+)@\w+([-.]\w+).\w+([-.]\w+)*`
评注：表单验证时很实用
匹配网址URL的正则表达式： `[a-zA-z]+://[^\s]*`
评注：网上流传的版本功能很有限，上面这个基本可以满足需求
匹配字符
. 匹配除换行符以外的任意字符
\w 匹配字母或数字或下划线或汉字
\s 匹配任意的空白符
\d 匹配数字
匹配位置
\b 匹配单词的开始或结束
^ 匹配字符串的开始
$ 匹配字符串的结束
\G 上一个匹配的结尾（本次匹配开始）
\A 字符串开头(类似^，但不受处理多行选项的影响)
\Z 字符串结尾或行尾(不受处理多行选项的影响)
\z 字符串结尾(类似$，但不受处理多行选项的影响)
重复

- 重复零次或更多次
- 重复一次或更多次

? 重复零次或一次
{n} 重复n次
{n,} 重复n次或更多次
{n,m} 重复n到m次
字符转义
如果想匹配元字符本身或者正则中的一些特殊字符，使用\转义。例如匹配*这个字符则使用*，匹配\这个字符，使用\。
需要转义的字符：$, (,), *, +, ., [,], ?, \, ^, {, }, |
字符类
当需要匹配明确的字符或字符集合时候，就用到字符类。
特殊字符
\0hh 8进制值hh所表示的字符
\xhh 16进制值hh所表示的字符
\uhhhh 16进制值hhhh所表示的Unicode字符
\t Tab
\n 换行符
\r 回车符
\f 换页符
\e Escape
\cN ASCII控制字符。比如\cC代表Ctrl+C
\p{name} Unicode中命名为name的字符类，例如\p{IsGreek}
陈列
[aeiou] 匹配一个元音字符
[.?!] 匹配给定的一个标点
范围
[0-9] 匹配0~9的数字，同\d
[a-z] 匹配所有小写字母
[a-zA-Z] 匹配所有字母
[a-z0-9A-Z_] 等同于\w
反义
表示不属于元字符或者字符类的字符
反义元字符
\W 匹配任意不是字母，数字，下划线，汉字的字符
\S 匹配任意不是空白符的字符
\D 匹配任意非数字的字符
\B 匹配不是单词开头或结束的位置

反义字符类
[^x] 匹配除了x以外的任意字符
[^aeiou] 匹配除了aeiou这几个字母以外的任意字符
分枝条件
又叫逻辑运算符，在此X和Y表示两个表达式
XY X紧跟Y
X|Y 表示X或Y，从左到右，满足第一个条件就不会继续匹配了。
分组
在这里我把表达式统一以\w为例：
(\w) 被一个括号包围起来是一个整体，表示一个分组
(\w)(\w) 自动命名分组，第一个小括号是分组1，第二个小括号是分组2
(?'Word'\w+)) 表示定义了一个叫做Word的分组
(?\w+)) 表示定义了一个叫做Word的分组
(?:\w+) 匹配exp,不捕获匹配的文本，也不给此分组分配组号
后向引用
后面的表达式可以引用前面的某个分组，用\1表示，就好像分组1的值赋值给了\1这个变量，这个变量可以在后面任意位置引用。
\1 表示分组1匹配的文本
\k 表示分组Word匹配的文本
匹配重复两个的英文，例如匹配Hello Hello、lei123 lei123：
(\w+)\s+\1
(?\w+)\s+\k
零宽断言（正向和负向）
零宽断言表示匹配字符的时候再添加一些定位条件，使匹配更精准。
\w+(?=ing) 匹配以ing结尾的多个字符（不包括ing）
\w+(?!ing) 匹配不是以ing结尾的多个字符
(?<=re)\w+ 匹配以re开头的多个字符（不包括re）
(?<!re)\w+ 匹配不是以re开头的多个字符
(?<=\s)\d+(?=\s) 匹配两边是空白符的数字，不包括空白符
贪婪与懒惰
贪婪：匹配尽可能长的字符串
懒惰：匹配尽可能短的字符串
懒惰模式的启用只需在重复元字符之后加?既可。
*? 重复任意次，但尽可能少重复
+? 重复1次或更多次，但尽可能少重复
?? 重复0次或1次，但尽可能少重复
{n,m}? 重复n到m次，但尽可能少重复
{n,}? 重复n次以上，但尽可能少重复
处理选项

在表达式里插记号的方式来启用绝大多数的模式，在正则的哪里插入，就从哪里启用。
(?i)：忽略大小写(CASE_INSENSITIVE)
(?x)：忽略空格字符(COMMENTS)
(?s)：.匹配任意字符，包括换行符（DOTALL）
(?m)：多行模式（MULTILINE）
(?u)：对Unicode符大小写不敏感（UNICODE_CASE），必须启用CASE_INSENSITIVE
(?d)：只有'\n'才被认作一行的中止（UNIX_LINES）
平衡组/递归匹配
平衡组用于匹配嵌套层次结构，常用于匹配HTML标签（当HTML内容不规范，起始标签和结束标签数量不同时，匹配出正确配对的标签），在此把表达式统一以\w为例。
(?'group'\w) 捕获的分组（\w匹配到的内容）命名为group，并压入堆栈
(?'-group'\w) 捕获分组（\w匹配到的内容）后，弹出group分组栈的栈顶内容（最后压入的捕获内容），堆栈本来为空，则本分组的匹配失败
(?(group)yes|no) 如果group栈非空匹配表达式yes，否则匹配表达式no
(?!) 零宽负向先行断言，由于没有后缀表达式，试图匹配总是失败