相当于查找和替换，但是比它们都要强大。

1、前期绑定：工具-引用-microsoft vbscript regular expressions 5.5可以看到关于正则的相关方法属性。

2、后期绑定：set regex=createobject("vbscript.regexp")

3、建议先用前期绑定，完成后采用后期绑定。

4、正则表达式测试工具使用。

|  |  |
| --- | --- |
|  |  |

Sub test()

Dim regx As New RegExp, m As Object, k As Object

sr = "aabcabcbac"

regx.Global = True 'Global属性：查找范围：true全部查找，false只查找第1个，默认false

regx.Pattern = "a" 'Pattern属性：书写正则表达式，默认为""

'----------------------------------

Set k = regx.Execute(sr) 'Execute方法：返回匹配成功的结果，是一个对象

For Each m In k

河南中医药大学龙子湖校区

Next

'----------------------------------

'----------------------------------

n = regx.Replace(sr, "-") 'Replace方法：将匹配成功的结果做替换。

MsgBox n

'----------------------------------

End Sub

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |
|  |  |  |

Sub shl()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

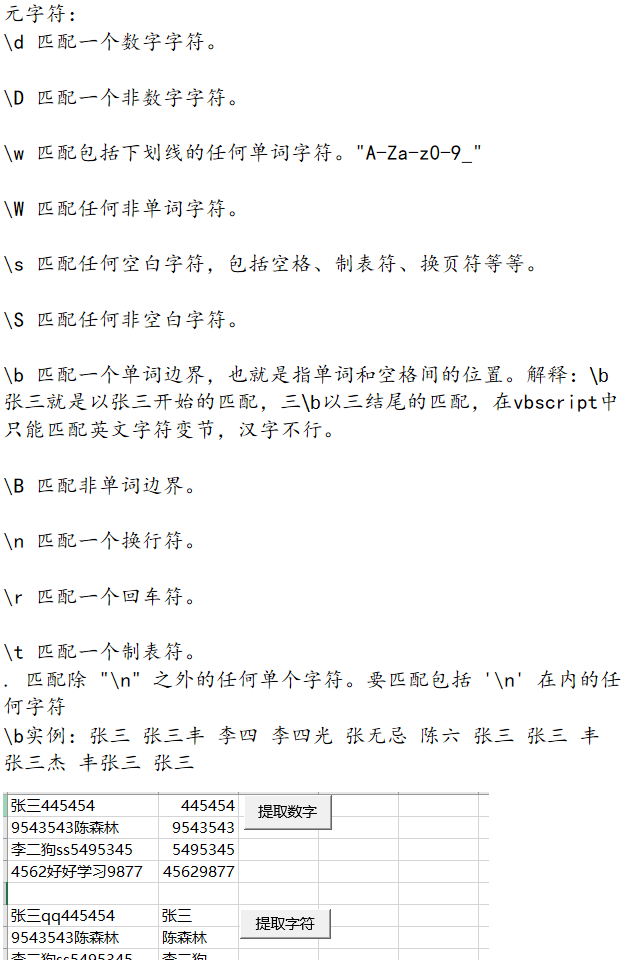
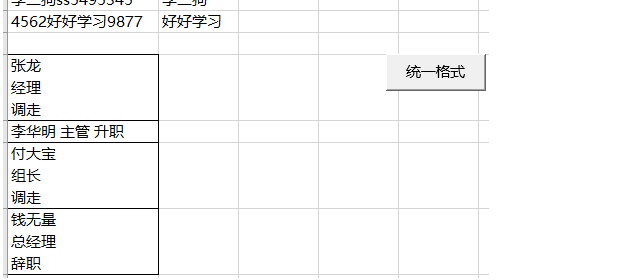
regex.Pattern = "消售"

For Each rng In Range("a2", Cells(Rows.Count, 1).End(xlUp))

rng = regex.Replace(rng.Value, "销售")

Next rng

End Sub

Sub shl()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.Pattern = "\D"

For Each rng In Range("a1:a4")

regex.Execute (rng.Value) '执行正则表达式的过程

rng(1, 2) = regex.Replace(rng.Value, "")

Next rng

End Sub

Sub slhl()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.Pattern = "\w" '解释在正则表达式工具测试器中，\w汉字也是匹配的，但是在vbscript中，汉字是不匹配的，只匹配A-Za-z0-9以及下划线。

For Each rng In [a6:a9]

rng(1, 2) = regex.Replace(rng.Value, "")

Next rng

End Sub

Sub shl()

Dim rng As Range, os As Object, o As Object

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.Pattern = "\s"

For Each rng In [a11:a14]

rng(1, 2) = regex.Replace(rng.Value, "-")

Next rng

End Sub

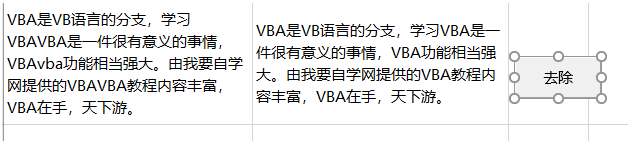
|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  | www+  成功  成功  成功  成功、匹配一次  不成功  不成功  不成功 |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  | 测试数据：  [www.baidu.com](http://www.baidu.com)  [www.51zxw.net](http://www.51zxw.net)  wwww.163..com  wwwwww.qq.com  w.haohao.cn  ww.uu.net  xuexi.cc |  |
|  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  | w{3,10}  成功  成功  成功，匹配一次  成功、匹配一次  不成功  不成功  不成功 |
|  | 测试数据：  [www.baidu.com](http://www.baidu.com)  [www.51zxw.net](http://www.51zxw.net)  wwww.163..com  wwwwww.qq.com  w.haohao.cn  ww.uu.net  xuexi.cc |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  | Sub shl()  Dim rng As Range, os As Object, o As Object  Set regex = CreateObject("vbscript.regexp")  regex.Global = True  regex.Pattern = "\S+"  For Each rng In [b1:b4]  Set os = regex.Execute(rng.Value)  For Each o In os  n = n + 1  rng(1, n + 1) = o.Value  Next o  n = 0  Next rng  End Sub |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  | Sub shl()  Dim rng As Range  Set regex = CreateObject("vbscript.regexp")  regex.Global = True  regex.Pattern = "-{2,}"  For Each rng In [a1:a6]  rng(1, 2) = regex.Replace(rng.Value, "--")  Next rng |
|  |  |  |
|  |  |  |



Sub shl()

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

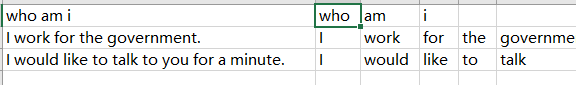
regex.ignorecase = True

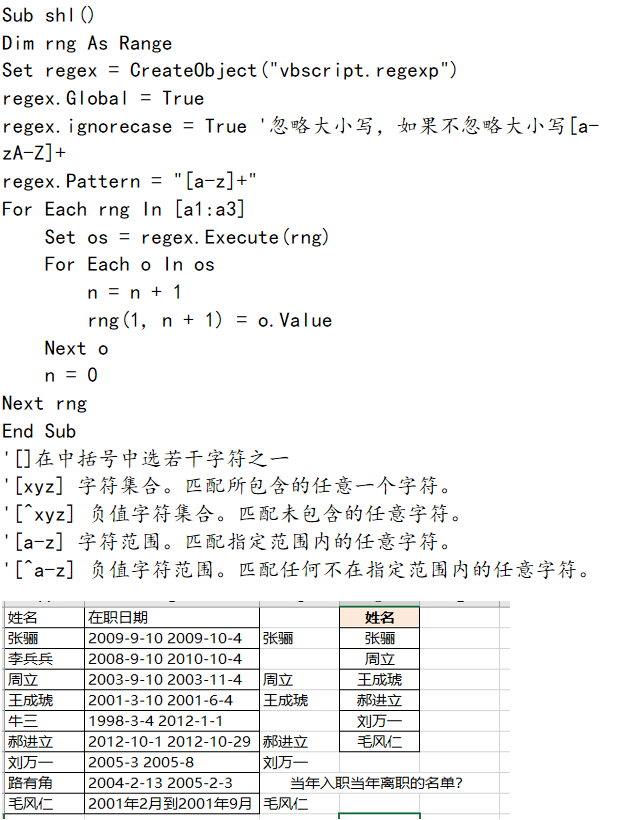
regex.Pattern = "(vba)+"

Range("b2").Value = regex.Replace([a1].Value, "VBA")

End Sub

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |
|  |  |  |





Sub shl()’分组

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

regex.Pattern = "(\d{4}).+\1.+"‘1代表第一个括号里面的，（）就等于分组

For Each rng In [b2:b10]

If regex.test(rng) Then‘判断正则表达式是否成立。

rng(1, 2) = rng(1, 0)

End If

Next rng

End Sub

'2.非捕获分组

'(?:pattern) 匹配 pattern 但不获取匹配结果，

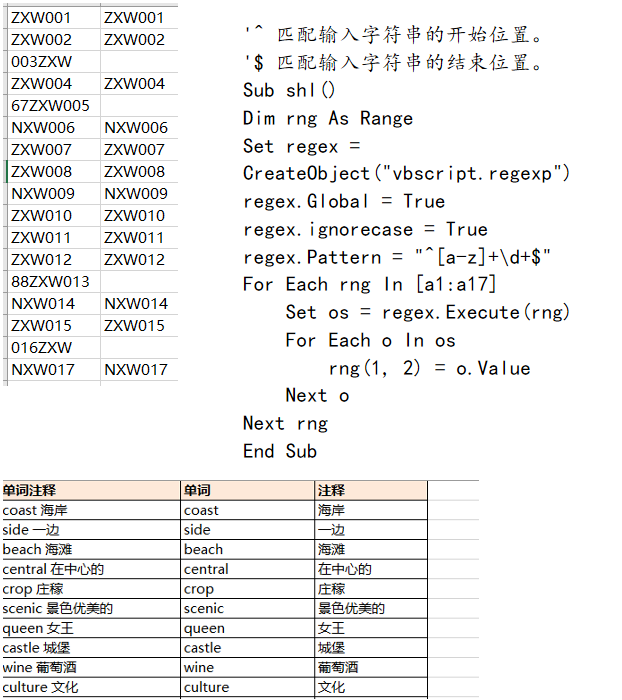
'也就是说这是一个非获取匹配，不进行存储供以后使用。

'这在使用 "或" 字符 (|) 来组合一个模式的各个部分是很有用。

'例如， 'industr(?:y|ies) 就是一个比 'industry|industries' 更简略的表达式。

'作用：避免不必要的捕获操作，提高代码的匹配效率。

'缺点：不美观，增加阅读难度。



正则表达式的循环引用

Sub shl()’以执行正则表达式的方式写入。

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

For Each ar In Array("[a-z -]+", "[^a-z -]+")‘正则表达式中汉字是不会提取的，只能反选。

n = n + 1

regex.Pattern = ar

For Each rng In Range("a2", Cells(Rows.Count, 1).End(xlUp))

Set os = regex.Execute(rng)

For Each o In os

rng(1, n + 1) = o.Value

Next o

Next rng

Next ar

End Sub

Sub shl1()‘以替换正则表达式的方式

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

For Each ar In Array("[^a-z -]+", "[a-z -]+")

n = n + 1

regex.Pattern = ar

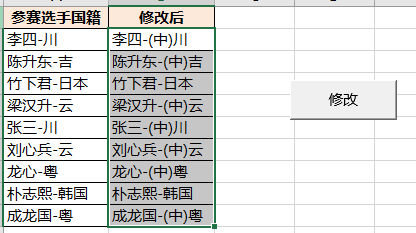
For Each rng In Range("a2", Cells(Rows.Count, 1).End(xlUp))

rng(1, n + 1) = regex.Replace(rng.Value, "")

Next rng

Next ar

End Sub



零宽断言只是在查找位置，而不是具体的字符匹配。vb中零宽断言不支持从右向左的匹配。

Sub shl()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

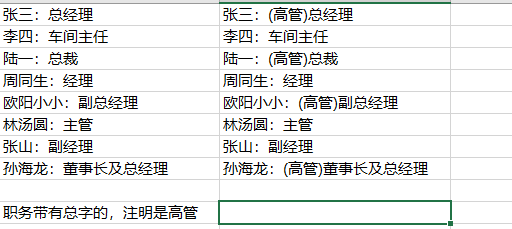
regex.Pattern = "(?=[川吉云粤])"‘正向零宽断言，只能查找前面的位置。

For Each rng In [a2:a10]

rng(1, 2) = regex.Replace(rng.Value, "(中)")

Next rng

End Sub



Sub shl()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

'regex.Pattern = "：(?=[副董]?总)" '"：(?=[副总])"在副和总在前都加上“（高管）”，会出现错误：没有总字，但是有副字也会加上。

regex.Pattern = "：(?=[\u4e00-\u9fa5]\*总)" '这个行不通，原因是因为总字前面只能有一个或没有字的情况下才能匹配成功。

For Each rng In [a1:a8]

rng(1, 2) = regex.Replace(rng.Value, "：(高管)")

Next rng

End Sub

Sub shl1()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

regex.Pattern = "(?=总)"

For Each rng In [a1:a8]

If regex.test(rng) Then

rng(1, 2) = regex.Replace(rng.Value, "(高管)")

End If

Next rng

regex.Global = True

regex.ignorecase = True

regex.Pattern = "(：?=[\u4e00-\u9fa5]+总)"

For Each rng In [a1:a8]

If regex.test(rng) Then

rng(1, 2) = regex.Replace(rng.Value, "：(高管)") '前面加：的原因是，零宽断言加“：”结合成为一个综合表达式，进行替换。

End If

Next rng

Dim rng1 As Range

For Each rng1 In [b1:b8]

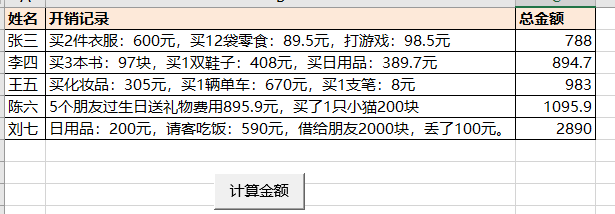
If rng1 = "" Then

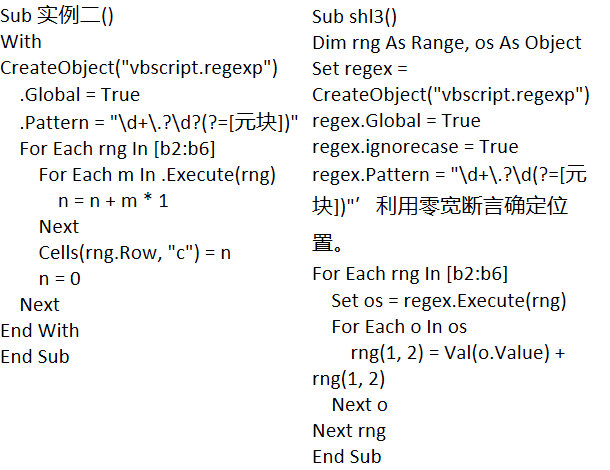
rng1.Value = rng1(1, 0).Value

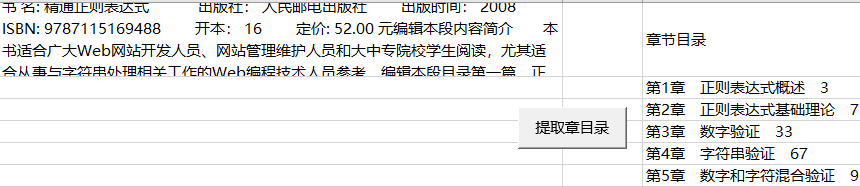
End If

Next rng1

End Sub







'前面我们已过知道"?"有三种：

'1 是量词{0,1}

'2 是非捕获型的匹配模式(?:)

'3 是环视结构(?=)(?!)

'今天学习第4种作用：

'当该字符紧跟在任何一个其他限制符 (\*, +, ?, {n}, {n,}, {n,m}) 后面时是懒惰模式

'懒惰模式：尽可能少的匹配所搜索的字符串

'贪婪模式：尽可能多的匹配所搜索的字符串（默认模式）

'

'

'

Sub test()

With CreateObject("vbscript.regexp")

.Global = True

.Pattern = "第\d+章.\*?[一-龢]+.+?\d+"

Set mat = .Execute([a1])

For Each m In mat

n = n + 1

Cells(n + 1, "c") = m

Next

End With

End Sub

自己写的代码

Sub shl()

Dim os As Object, o As Object

Set regex = CreateObject("vbscript.regexp")

With regex

.Global = True

.ignorecase = True

.Pattern = "第\d+章.{0,}?[\u4e00-\u9fa5]+.{0,}?\d+"这个表达式是可以简化的，简化成“第\d+章.\*？\d+”一样能匹配出所有结果；原因在于表达式前面有限定，后面也有限定，这个时候中间是什么无所谓，只要匹配出在前后都有限定的情况下最小的字符串就行。这才是真正懒惰模式的应用。懒惰模式中“.”的用途很大。前后限定好了，这个基本上是通用的，处理换行符不能匹配之外，所有的都成立。

Set os = .Execute([a1])

For Each o In os

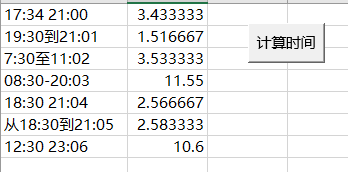
n = n + 1

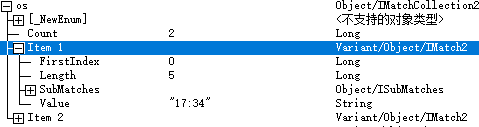
Cells(n + 1, "e") = o.Value

Next o

End With

End Sub





os对象的属性，firstindex表示匹配的第一个字符的起始位置。

length表示匹配的字符串长度。

Sub shl()

Dim rng As Range, os As Object, o As Object

Set regex = CreateObject("vbscript.regexp")

With regex

.Global = True

.ignorecase = True

.MultiLine = False默认是false表示多行匹配，true为单行匹配。

.Pattern = "\d+:\d+"

For Each rng In [a1:a7]

Set os = .Execute(rng)

For Each o In os

n = n + 1

rng(1, n + 2) = o.Value

Next o

rng(1, 2) = (rng(1, 4) - rng(1, 3)) \* 24

n = 0

Next rng

End With

End Sub

Sub lsf()

Dim rng As Range, os As Object

Set regex = CreateObject("vbscript.regexp")

With regex

.Global = True

.ignorecase = True

.Pattern = "\d+:\d+"

For Each rng In [a1:a7]

Set os = .Execute(rng.Value)

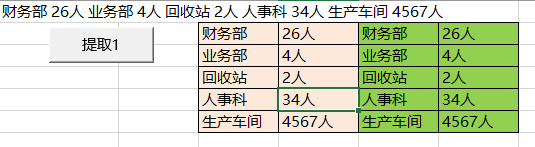
rng(1, 2).Value = (CDate(os(1).Value) - CDate(os(0).Value)) \* 24

‘cdate表示转化为时间格式，\*24表示原本以天为单位，改为以小时为单位，如果是\*24\*60表示以分钟为单位。

Next rng

End With

End Sub



Sub shl()

Dim o As Object, os As Object

Set regex = CreateObject("vbscript.regexp")

With regex

.Global = True

.ignorecase = True

.Pattern = "([\u4e00-\u9fa5]+) (\d+人)"

Set os = .Execute([a1])

For Each o In os

n = n + 1

Cells(n + 1, 3) = .Replace(o.Value, "$1")’$1表示匹配后分组中的第一个组

Cells(n + 1, 4) = .Replace(o.Value, "$2")

Next o

End With

End Sub

Sub shl1()

Dim o As Object, os As Object

Set regex = CreateObject("vbscript.regexp")

With regex

.Global = True

.ignorecase = True

.MultiLine = True

.Pattern = "([\u4e00-\u9fa5]+) (\d+人)"

Set os = .Execute([a1])

' For Each o In os

' n = n + 1

' Cells(n + 1, 5) = o.submatches(0)

' Cells(n + 1, 6) = o.submatches(1)

' Next o

For i = 0 To os.Count - 1

n = n + 1

Cells(n + 1, 5) = os(i).submatches(0)

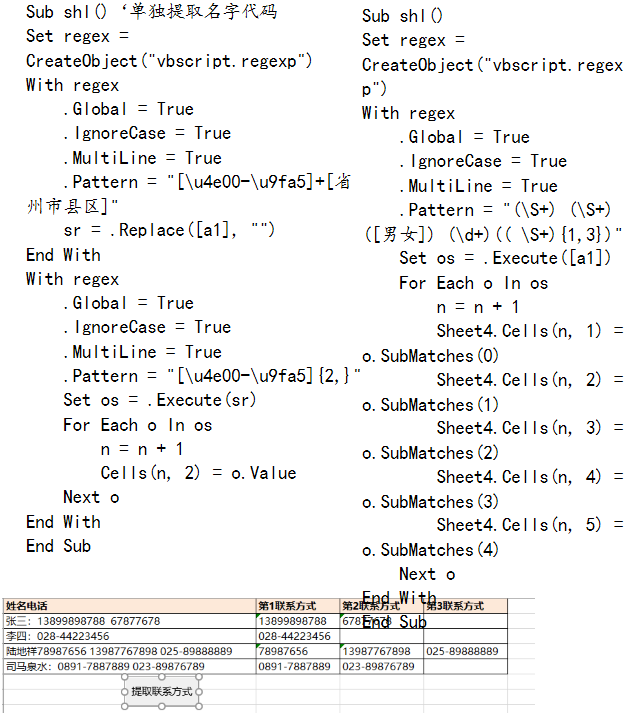
Cells(n + 1, 6) = os(i).submatches(1)

Next i

End With

End Sub

|  |
| --- |
| 苗男萌 420111196405251685 女 48 湖北省 武汉市 洪山区 薛曼女 44081119680302798X 女 44 广东省 湛江市 麻章区 于鑫 542624197710246658 男 35 西藏自治区 林芝地区 墨脱县 融玲翡 150824198306256601 女 29 内蒙古自治区 顾君 530422198409094752 男 28 云南省 玉溪市 澄江县 钱嘉男 469035196202167193 男 50 海南省 麻修勇 511527197704238990 男 35 四川省 宜宾市 筠连县 边涵蕾 530622197512261287 女 37 云南省 卜微 350200196811269483 女 44 福建省 厦门市 厦门市 阚小乐 411600197411288734 男 38 河南省 融碧蓉 451028199009293463 女 22 广西壮族自治区 廉代卉 44010319680306028X 女 44 广东省 广州市 荔湾区 朱争 210726198703014378 男 25 辽宁省 锦州市 黑山县 徐贝 610601196401187807 女 48 陕西省 延安市 市辖区 殷冬 430781198411248493 男 28 湖南省 常德市 津市市 孙凌春 431124198402078462 女 28 湖南省 永州市 道县 丁红梅 130981198206188284 女 30 河北省 沧州市 泊头市 蔚然 632324196704122182 女 45 青海省 黄南藏族自治州 河南蒙古族自治县 宓香菱 371103196505169263 女 47 山东省 日照市 |



Sub shl()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

regex.Pattern = "[0-9-]+" '"\d+-?\d+"

For Each rng In [a2:a5]

Set os = regex.Execute(rng)

For Each o In os

n = n + 1

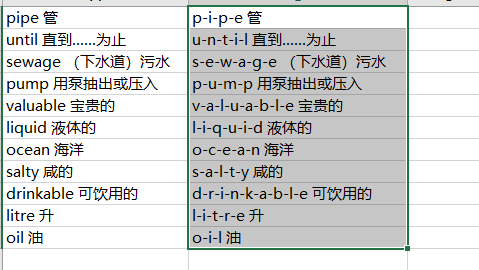
rng(1, n + 1) = o.Value

Next o

n = 0

Next rng

End Sub



Sub shl()

Dim rng As Range

Set regex = CreateObject("vbscript.regexp")

regex.Global = True

regex.ignorecase = True

regex.Pattern = "(?!^)(?=\w)"

For Each rng In Range("a1", Cells(Rows.Count, 1).End(xlUp))

rng(1, 2) = regex.Replace(rng.Value, "-")

Next rng

End Sub

