Section 10

Regular Expressions

Mohammed Asir Shahid

2021-08-03

Contents

1 Regular Expression Basics

In this lesson, we will be working with pattern matching and regular expressions. Regular expressions allow you to specify a pattern of text to search for.

An example of a text pattern would be a phone number. 415-555-0000. In the US, this is the standard way of writing up phone numbers. If we had that same number but without the hyphens, we would not recognize it as a phone number.

```
def isPhoneNumber(text):
    if len(text) != 12:
        return False
    for i in range(0, 3):
        if not text[i].isdecimal():
            return False
    if text[3] != '-':
        return False
    for i in range(4, 7):
```

```
        if not text[i].isdecimal():
            return False
    if text[7] != '-':
        return False
    for i in range(8, 12):
        if not text[i].isdecimal():
            return False
    return True

print(isPhoneNumber("415-555-1234"))
print(isPhoneNumber("My Phone Number"))
```

```
True
False
```

That's a lot of code for a relatively simple task. If we want to find phone numbers in large strings, we'd need to write some more code.

```
def isPhoneNumber(text):
    if len(text) != 12:
        return False
    for i in range(0, 3):
        if not text[i].isdecimal():
            return False
    if text[3] != '-':
        return False
    for i in range(4, 7):
        if not text[i].isdecimal():
            return False
    if text[7] != '-':
        return False
    for i in range(8, 12):
        if not text[i].isdecimal():
            return False
    return True

print(isPhoneNumber("415-555-1234"))
print(isPhoneNumber("My Phone Number"))
```

```
message="Call me at 415-555-1011 tomorrow, or 415-555-9999 for my office line"

foundNuumber=False

for i in range(len(message)):
    chunk=message[i:i+12]
    if isPhoneNumber(chunk) == True:
        print("Phone number found")
        foundNumber=True
if not foundNumber:
    print("Could not find any phone numbers")



True
False
Phone number found
Phone number found
```

1.1 The re Module

We can write the previous code much faster using regular expressions.

```
import re

message="Call me at 415-555-1011 tomorrow, or 415-555-9999 for my office line"

phoneNumRegex=re.compile(r"\d\d\d-\d\d\d-\d\d\d\d")

mo=phoneNumRegex.search(message)

print(type(mo))
print(mo.group())

print(phoneNumRegex.findall(message))



<class 're.Match'>
415-555-1011
['415-555-1011', '415-555-9999']
```

2 Regex Groups and the Pipe Character

We can try some more of Python's more powerful pattern matching capabilities.

Let's say we want to seperate the area code from a phone number.

```
import re

phoneNumRegex = re.compile(r"\d\d\d-\d\d\d-\d\d\d\d")
mo=phoneNumRegex.search("My number is 415-555-4242")
print(mo.group())
```

```
415-555-4242
```

2.1 Groups

Let's say we only want the phone number or only the phone number portion of the number. We can do this sing parentheses to mark out groups in the string.

```
import re

phoneNumRegex = re.compile(r"(\d\d\d)-(\d\d\d-\d\d\d\d)")
mo=phoneNumRegex.search("My number is 415-555-4242")
print(mo.group())

print(mo.group(1))
print(mo.group(2))
```

```
415-555-4242
415
555-4242
```

The parentheses there can be useful syntax when we want to find specific parts of something. However, what can we do when we want to find literal parentheses? We would escape them using parentheses.

```
import re

phoneNumRegex = re.compile(r"\(\d\d\d\) \d\d\d-\d\d\d\d")
mo=phoneNumRegex.search("My number is (415) 555-4242")
print(mo.group())
```

```
(415) 555-4242
```

2.2   Pipe Character |

Pipes can be used to match one of several patterns as part of the regular
expression.

Let's say we wanted to match any of the strings "Batman", "Batmobile",
"Batcopter", or "Batbat"

```
import re

batRegex = re.compile(r"Bat(man|mobile|copter|bat)")
mo=batRegex.search("Batmobile lost a wheel.")
print(mo.group())
print(mo.group(1))
```

```
Batmobile
mobile
```

If the search method can't find the regular expression pattern, it will
return None. In that case, we can risk running into errors.