Lecture 28: Reg Ex Problems

Hexadecimal Colors

Write a regular expression to match a hexadecimal color value in a piece of text. A hexadecimal color value is a 6 character sequence where each character is a hexadecimal digit (i.e. between 0 and f) preceded by an optional #. For example `#ff34d5` is valid but `#h56732` is not. Make sure to group the actual hex number for ease-of-use.

Hexadecimal Colors

Write a regular expression to match a hexadecimal color value in a piece of text. A hexadecimal color value is a 6 character sequence where each character is a hexadecimal digit (i.e. between 0 and f) preceded by an optional #. For example `#ff34d5` is valid but `#h56732` is not. Make sure to group the actual hex number for ease-of-use.

```
#?({0-9a-f}{6})
```

IP Addresses

IP addresses are strings of four numbers, delimited by a period, where each number is in the range $[0, 255]$. For example, the IP address of this computer is 137.165.206.66. The IP address for the Google Domain Name Server is 8.8.8.8, which can also be written as 8.08.008.8. Write a regular expression to check if some text is exactly an IP address. That is, do IP address validation.

IP Addresses

IP addresses are strings of four numbers, delimited by a period, where each number is in the range $[0, 255]$. For example, the IP address of this computer is 137.165.206.66. The IP address for the Google Domain Name Server is 8.8.8.8, which can also be written as 8.08.008.8. Write a regular expression to check if some text is exactly an IP address. That is, do IP address validation.

```
^(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$
```

IP Addresses

IP addresses are strings of four numbers, delimited by a period, where each number is in the range $[0, 255]$. For example, the IP address of this computer is 137.165.206.66. The IP address for the Google Domain Name Server is 8.8.8.8, which can also be written as 8.08.008.8. Write a regular expression to check if some text is exactly an IP address. That is, do IP address validation.

```
^(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$
```

And here's a way to programmatically create the regular expression:

```
ips = []
for i in range(256):
    if (i < 10):
        ips.append(str(i).zfill(2))
    if (i < 100):
        ips.append(str(i).zfill(3))
    ips.append(str(i))

regexips ="^(({0})\\.){{3}}({0})$".format("|".join(num for num in ips))
```

Email Addresses

Write a regular expression to check whether some given text is a *valid* email address. A valid email address may contain the characters `.`, `%`, `+`, and `-`. Suppose, incorrectly, that all email addresses must end with a a 2-4 character string.

Email Addresses

Write a regular expression to check whether some given text is a *valid* email address. A valid email address may contain the characters `.`, `%`, `+`, and `-`. Suppose, incorrectly, that all email addresses must end with a a 2-4 character string.

```
^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$
```