Regular Expression

Character types

- `\w` matches with any alphanumeric character
- `.` matches to all characters, including symbols
- `\d` matches to all single digits [0-9]
- `\s` matches to all single spaces
- `\.` matches to the period character

```
import re
re.findall("\w", "h32rb17")


import re
re.findall("\d", "h32rb17")
```

Quantify occurrences

- `+` symbol represents one or more occurrences of a specific character.
- `*` symbol represents zero, one, or more occurrences of a specific character.
- `\d{2}` instructs Python to return all matches of exactly two single digits
- `\d{1,3}` ?? 1 - 3 ??

```
import re
re.findall("\d+", "h32rb17")


import re
re.findall("\d*", "h32rb17")


import re
re.findall("\d{2}", "h32rb17 k825t0m c2994eh")


import re
re.findall("\d{1,3}", "h32rb17 k825t0m c2994eh")
```

```
import re
pattern = "\w+:\s\d+"
```

```
employee_logins_string = "1001 bmoreno: 12 Marketing 1002 tshah: 7 Human Resources 1003 sgilmore: 5 Finance"
print(re.findall(pattern, employee_logins_string))
```

```
['bmoreno: 12', 'tshah: 7', 'sgilmore: 5']
```

IP addr.

```
# Assign `log_file` to a string containing username, date, login time, and IP address for a series of login attempts
log_file = "eraab 2022-05-10 6:03:41 192.168.152.148 \niuduike 2022-05-09 6:46:40 192.168.22.115 \nsmartell 2022-05-09 19:30:32 192.168.190.178 \narutley 2022-05-12 17:00:59 1923.1689.3.24 \nrjensen 2022-05-11 0:59:26 192.168.213.128 \naestrada 2022-05-09 19:28:12 1924.1680.27.57 \nasundara 2022-05-11 18:38:07 192.168.96.200 \ndkot 2022-05-12 10:52:00 1921.168.1283.75 \nabernard 2022-05-12 23:38:46 19245.168.2345.49 \ncjackson 2022-05-12 19:36:42 192.168.247.153 \njclark 2022-05-10 10:48:02 192.168.174.117 \nalevitsk 2022-05-08 12:09:10 192.16874.1390.176 \njrafael 2022-05-10 22:40:01 192.168.148.115 \nyappiah 2022-05-12 10:37:22 192.168.103.10654 \ndaquino 2022-05-08 7:02:35 192.168.168.144"

# Assign `pattern` to a regular expression that matches with all valid IP addresses and only those
pattern = "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}"

# Use `re.findall()` on `pattern` and `log_file` and assign `valid_ip_addresses` to the output
valid_ip_addresses = re.findall(pattern, log_file)

# Assign `flagged_addresses` to a list of IP addresses that have been previously flagged for unusual activity
flagged_addresses = ["192.168.190.178", "192.168.96.200", "192.168.174.117", "192.168.168.144"]

# Iterative statement begins here
# Loop through `valid_ip_addresses` with `address` as the loop variable
for address in valid_ip_addresses:

    # Conditional begins here
    # If `address` belongs to `flagged_addresses`, display "The IP address _____ has been flagged for further analysis."
    if address in flagged_addresses:
        print("The IP address", address, "has been flagged for further analysis.")

    # Otherwise, display "The IP address _____ does not require further analysis."
    else:
```

```
    print("The IP address", address, "does not require further analysis.")
```

Revision #4
Created 22 September 2024 11:44:16 by Admin
Updated 22 September 2024 12:28:15 by Admin