Marwadi University

01CE1705-Programming with Python

# Unit-7 Regular Expression

Prof. Chetan Chudasama
Computer Engineering Department

# Regular Expression


- It is used for pattern matching.
- It is an expression that is used to extract information.
- Python has a built-in module called re, which can be used to work with Regular Expressions.re module is a collection of predefined functions that is used to process the input text.

**Applications:-**

- Regular expression is used for validation.
- It is used to develop pattern matching application.
- To develop translators like compiler,interpreters,assemblers .
- To develop communication protocols TCP/IP,UDP,HTTPS.

# Character Classes


| Set | Description |
|---|---|
| [abc] | either a or b or c |
| [^abc] | Except a and b and c |
| [a-z] | Any lowercase alphabet |
| [A-Z] | Any uppercase alphabet |
| [a-zA-Z] | Any alphabet |
| [0-9] | Any digit |
| [a-zA-Z0-9] | Any alphanumeric character |
| [^a-zA-Z0-9] | Any character(special character) except alphanumeric character |

# Predefined Character Classes

Marwadi University

| Character | Description |
|---|---|
| \s | space character |
| \S | except space character |
| \d | Any digit |
| \D | except digits |
| \w | Any word character(alpha numeric character) |
| \W | Any character except word(special character) |
| .(dot) | Every character |

# Metacharacters

Marwadi University

| Character | Description |
|---|---|
| [] | A set of character |
| \ | can also be used to escape special characters |
| ^ | starts with |
| $ | ends with |
| * | Zero or more occurrences |
| + | One or more occurrences |
| ? | Zero or one occurrences |
| {} | Exactly the specified number of occurrences |
| \| | Either Or |
| () | Capture and group |

# Regex Functions


**match():-**

- It is used to test the input string starts with specified pattern or not.
- On success, it returns the match object and on failure, returns None.

Example:-

```
import re
str1=input("Enter pattern to check ")
m=re.match(str1,'abcdefgh')
if m!=None:
    print('Match is available at beginning of the string')
else:
    print('Match is not available at beginning of the string')
```

Example:-

```
import re
target_string="virat kohli was born on 5th Nov"
result=re.match('\w{5}',target_string)
print("Match object:",result)
print("Match value:",result.group())
```

**fullmatch():-**

- It returns a match object if and only if the entire string matches the pattern.Otherwise, it will return None.

Example:-

```
import re
str1="marwadi university rajkot"
print(re.match("marwadi",str1))
print(re.fullmatch("marwadi",str1))
```

**search():-**

- It is used to test the specified pattern is present or not in the given string.
- It returns None (if the pattern doesn’t match), or a re.MatchObject that contains information about the matching part of the string.
- This method stops after the first match.

Example:-

```
import re
test_string="marwadi university rajkot university"
s=re.search("university",test_string)
print(s)
```

**findall():-**

- It returns a list containing all matches.
- The list contains the matches in the order they are found.
- If no matches are found, an empty list is returned.

Example:-

```
import re
test_string="marwadi university rajkot university"
list1=re.findall("university",test_string)
print(list1)
```

Example:-

```
import re
lst=re.findall('[0-9]','a7b9k6z')
print(lst)
```

**sub():-**

- It replaces the matches with the text of your choice.

•We can control the number of replacements by specifying the count parameter

Example:-

```
import re
result=re.sub('\d','#','a7b9k5t9k',2)
print(result)
```

**subn():-**

- This method is similar to sub() and also returns new string along with number of replacements.
- It return tuple(consist of new string and number of replacement).

Example:-

```
import re
str1=re.subn('\d','#','a7b9k5t9k')
print("The result string is:",str1[0])
print("The number of replacements“,str1[1])
```

**split():-**

- This function splits the given string according to the occurrence of a particular character or pattern.
- Upon finding the pattern, this function returns the remaining characters from the string in a list.

Example:-

```
import re
result=re.split('-','10-20-30-40-50')
print(result)
```

```
lst=re.split('\.','www.marwadiuniversity.ac.in')
print(lst)
for x in lst:
    print(x)
```

**compile():-**

- It is regular expression pattern into pattern object, which can be used for pattern matching.
- It also help to search a pattern again without rewriting it.

Example:-

```
import re
pattern=re.compile("\d{3}")
print(pattern.findall("virat's lucky number is 183 018 251"))
print(pattern.findall("rohit's lucky number is 45 209"))
```

# Examples


**Example:1 Mobile number validation**

```
import re
s=input("Enter mobile number ")
m=re.fullmatch('(\+91)[6-9]{1}[0-9]{9}',s)
if m!=None:
    print(s,'is valid identifier')
else:
    print(s,'is not valid identifier’)
```

**Example:2 Email validation**

```
import re
s=input("Enter mail id ")
m=re.fullmatch("[a-zA-Z][a-zA-Z0-9]+[._]?[a-zA-Z0-9]+@[a-zA-Z]{5,10}\.(com|in)",s)
if m!=None:
    print("Valid mail id")
else:
    print("Invalid mail id")
```

**Example:3 Extract all mobile number from text file and store it into another file**

```
import re
f1=open("input.txt","r")
f2=open("output.txt","w")
for line in f1:
    list1=re.findall('[6-9][0-9]{9}',line)
    for number in list1:
        f2.write(number+"\n")
f2.close()
f1.close()
```

**Example:4 Web Scraping**

```
import re, urllib, urllib.request
s=urllib.request.urlopen("https://www.w3schools.com/python/ref_list_append.asp")
text=s.read()
print(text)
m1=re.findall("[0-9]{14}",str(text))
print(m1)
```

Marwadi University

# Thank You