

PRACTICAL VI

DATE: 12/03/24

Regular Expressions

AIM: To learn and execute Regular Expressions in python.

INTRODUCTION:

A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern.

RegEx can be used to check if a string contains the specified search pattern.

RegEx Module

Python has a built-in package called re, which can be used to work with Regular Expressions.

Import the re module:

```
import re
```

RegEx Functions

The re module offers a set of functions that allows us to search a string for a match:

Function	Description
findall	Returns a list containing all matches
search	Returns a Match object if there is a match anywhere in the string
split	Returns a list where the string has been split at each match
sub	Replaces one or many matches with a string

Metacharacters

Metacharacters are characters with a special meaning:

Character	Description
-----------	-------------

[]	A set of characters
\	Signals a special sequence (can also be used to escape special characters)
.	Any character (except newline character)
^	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

Special Sequences

A special sequence is a \ followed by one of the characters in the list below, and has a special meaning:

Character	Description
\A	Returns a match if the specified characters are at the beginning of the string
\b	Returns a match where the specified characters are at the beginning or at the end of a word (the "r" in the beginning is making sure that the string is being treated as a "raw string")

\B	Returns a match where the specified characters are present, but NOT at the beginning (or at the end) of a word (the "r" in the beginning is making sure that the string is being treated as a "raw string")
\d	Returns a match where the string contains digits (numbers from 0-9)
\D	Returns a match where the string DOES NOT contain digits
\s	Returns a match where the string contains a white space character
\S	Returns a match where the string DOES NOT contain a white space character
\w	Returns a match where the string contains any word characters (characters from a to Z, digits from 0-9, and the underscore _ character)
\W	Returns a match where the string DOES NOT contain any word characters
\Z	Returns a match if the specified characters are at the end of the string

Sets

A set is a set of characters inside a pair of square brackets [] with a special meaning:

Set	Description
[arn]	Returns a match where one of the specified characters (a, r, or n) is present
[a-n]	Returns a match for any lower case character, alphabetically between a and n
[^arn]	Returns a match for any character EXCEPT a, r, and n
[0123]	Returns a match where any of the specified digits (0, 1, 2, or 3) are present
[0-9]	Returns a match for any digit between 0 and 9
[0-5][0-9]	Returns a match for any two-digit numbers from 00 and 59

[a-zA-Z]	Returns a match for any character alphabetically between a and z, lower case OR upper case
[+]	In sets, +, *, ., , (), \$, {} has no special meaning, so [+] means: return a match for any + character in the string

Q1. Write a Python program to check that a string contains only a certain set of characters (in this case a-z, A-Z and 0-9).

```
import re

string = input("Enter a string:")

if(re.search("^[a-zA-Z0-9]+$",string)):
    print("Pattern matched")
else:
    print("Pattern did not match")
```



```
Enter a string:Anitya24
Pattern matched
```

Figure 1: Output of Q1

Q2. Write a Python program to remove leading zeros from an IP address.

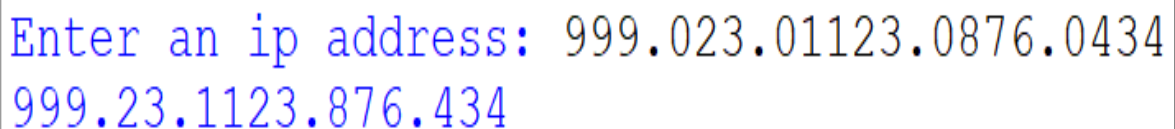
```
import re

ip = input("Enter an ip address: ")

string = re.sub('\.[0]*', '.', ip)

string1 = re.sub('^[0]*', '', string)

print(string1)
```

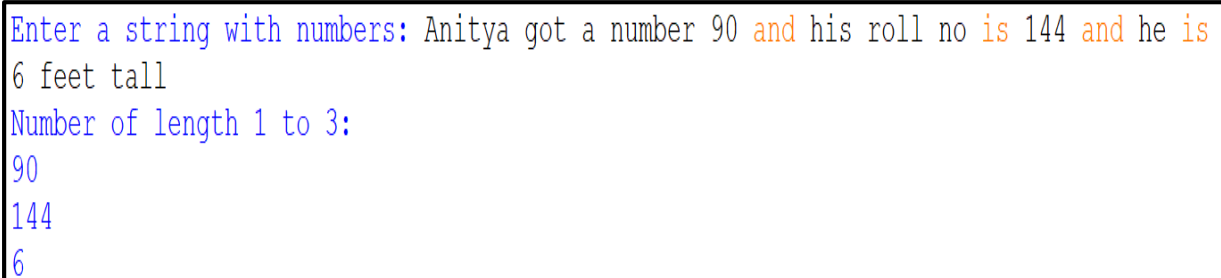


```
Enter an ip address: 999.023.01123.0876.0434
999.23.1123.876.434
```

Figure 2: Output of Q2

Q3. Write a Python program to search the numbers (0-9) of length between 1 to 3 in a given string.

```
import re
reg = input("Enter a string with numbers: ")
results = re.finditer(r"([0-9]{1,3})", reg)
print("Number of length 1 to 3: ")
for n in results:
    print(n.group(0))
```

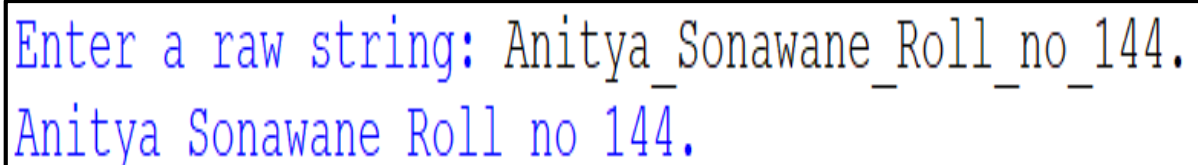
A screenshot of a terminal window showing the execution of a Python program. The input string is "Anitya got a number 90 and his roll no is 144 and he is 6 feet tall". The program searches for numbers with lengths between 1 and 3. The output shows the numbers 90, 144, and 6, each on a new line.

```
Enter a string with numbers: Anitya got a number 90 and his roll no is 144 and he is
6 feet tall
Number of length 1 to 3:
90
144
6
```

Figure 3: Output of Q3

Q4. Write a Python program to replace whitespaces with an underscore and vice versa.

```
import re
string = input("Enter a raw string: ")
new_str = re.sub('_', ' ', string)
print(new_str)
```

A screenshot of a terminal window showing the execution of a Python program. The input string is "Anitya Sonawane Roll no 144.". The program replaces spaces with underscores. The output shows the string "Anitya_Sonawane_Roll_no_144.".

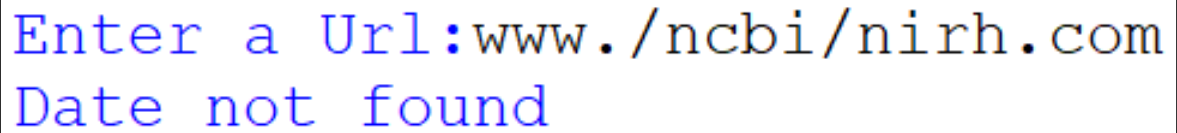
```
Enter a raw string: Anitya Sonawane Roll no 144.
Anitya_Sonawane_Roll_no_144.
```

Figure 4: Output of Q4

Q5. Write a Python program to extract year, month and date from an URL.

```
import re
Url = input("Enter a Url:")
x = re.findall('(\d{4})-(\d{1,2})-(\d{1,2})', Url)
if x:
```

```
print("Date found :",x)
else:
    print("Date not found ")
```

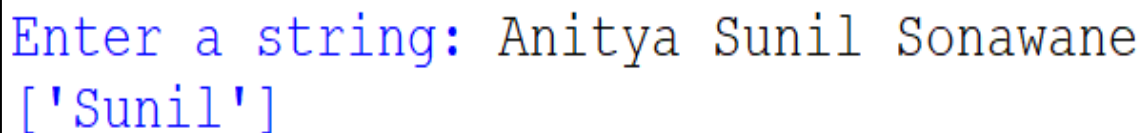
A screenshot of a terminal window showing the output of a Python program. The first line is 'Enter a Url:www./ncbi/nirh.com' and the second line is 'Date not found'. Both lines are displayed in a blue monospaced font.

```
Enter a Url:www./ncbi/nirh.com
Date not found
```

Figure 5: Output of Q5

Q6. Write a Python program to find all five characters long word in a string.

```
import re
text = input("Enter a string: ")
print(re.findall(r"\b\w{5}\b", text))
```

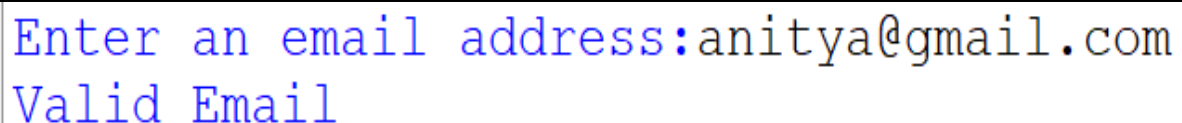
A screenshot of a terminal window showing the output of a Python program. The first line is 'Enter a string: Anitya Sunil Sonawane' and the second line is ['Sunil']. Both lines are displayed in a blue monospaced font.

```
Enter a string: Anitya Sunil Sonawane
['Sunil']
```

Figure 6: Output of Q6

Q7. Write a Python regex code for email validation.

```
import re
email = input("Enter an email address")
regex = '^([a-z0-9]+[\.\_]?[a-z0-9]+[@]\w+[.]\w{2,3})$'
if(re.search(regex,email)):
    print("Valid Email")
else:
    print("Invalid Email")
```

A screenshot of a terminal window showing the output of a Python program. The first line is 'Enter an email address:anitya@gmail.com' and the second line is 'Valid Email'. Both lines are displayed in a blue monospaced font.

```
Enter an email address:anitya@gmail.com
Valid Email
```

Figure 7: Output of Q7