PYTHON DAY - 10

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

REGULAR EXPRESSION

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

REG EX

- A regular expression, often abbreviated as regex, is a pattern of characters that is used to match and manipulate strings of text.

- It's a powerful tool for searching, validating, and manipulating text data

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

REG EX

Reg ex

```
import re
```

FUNCTIONS IN REG EX

Reg ex

```
re.search(pattern, text)

re.findall(pattern, text)

re.sub(pattern, sub, text)

re.fullmatch(pattern, text)
```

PATTERN

- In regular expressions (regex), a pattern is a sequence of characters and metacharacters that define a specific search pattern.

- The pattern is used to match and search for specific patterns in a larger string.

- These patterns can be combined and modified to create more complex expressions that can match a wide range of strings.

ANCHORS

Anchors	
^	Start of line +
\A	Start of string +
$	End of line +
\Z	End of string +
\b	Word boundary +
\B	Not word boundary +
\<	Start of word
\>	End of word

CHARACTER CLASSES

Character Classes	
\c	Control character
\s	White space
\S	Not white space
\d	Digit
\D	Not digit
\w	Word
\W	Not word
\xhh	Hexadecimal character hh
\Oxxx	Octal character xxx

RANGES

Ranges	
.	Any character except new line (\n) +
(a|b)	a or b +
(...)	Group +
(?:...)	Passive Group +
[abc]	Range (a or b or c) +
[^abc]	Not a or b or c +
[a-q]	Letter between a and q +
[A-Q]	Upper case letter + between A and Q +
[0-7]	Digit between 0 and 7 +
\n	nth group/subpattern +

QUANTIFIERS

Quantifiers	
*	0 or more +
*?	0 or more, ungreedy +
+	1 or more +
+?	1 or more, ungreedy +
?	0 or 1 +
??	0 or 1, ungreedy +
{3}	Exactly 3 +
{3,}	3 or more +
{3,5}	3, 4 or 5 +
{3,5}?	3, 4 or 5, ungreedy +

SAMPLES

Sample Patterns	
([A-Za-z0-9-]+)	Letters, numbers and hyphens
(\d{1,2}\/\d{1,2}\/\d{4})	Date (e.g. 21/3/2006)
([^\s]+(?=\.(jpg|gif|png))\.\2)	jpg, gif or png image
(^[1-9]{1}$\|^[1-4]{1}[0-9]{1}$|^50$)	Any number from 1 to 50 inclusive
(#?([A-Fa-f0-9]){3}(([A-Fa-f0-9]){3})?)	Valid hexadecimal colour code
((?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,15})	8 to 15 character string with at least one upper case letter, one lower case letter, and one digit (useful for passwords).
(\w+@[a-zA-Z_]+?\.[a-zA-Z]{2,6})	Email addresses
(\<(/?[^\>]+)\>)	HTML Tags

NUMBERS, LETTERS, HYPHENS

```python
import re


pattern = '([a-zA-Z0-9\-]+)'


test_string = 'hello-world-123'


result = re.search(pattern, test_string)
```

DATE

```
import re


pattern = '(\d{1,2}\/\d{1,2}\/\d{4})'


test_string = '21/3/2006'


result = re.search(pattern, test_string)
```

FINDING FILE NAME WITH EXTENSIONS

```python
import re


pattern = '[^\s]+(?=\.(png|jpg|gif))'


test_string = 'image.png'


result = re.search(pattern, test_string)
```

FINDING HEXADECIMAL COLOUR CODE

```python
import re


pattern = '#([A-Fa-f0-9]{6}|[A-Fa-f0-9]{3})'


test_string = '#FFA500'


result = re.search(pattern, test_string)
```

PASSWORD

```
pattern = r'\b(?=\S{6,})(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*\W)[^\s]{6,}\b'


sentence = "Abc@12345"


result = re.search(pattern, sentence)
```

EMAIL

```
pattern =  r"[a-zA-Z0-9._%+-]*@[a-zA-Z0-9.-]+\.[a-zA-Z]{3,}"


sentence = "abcA23_@gmail.com"


result = re.search(pattern, sentence)
```

HTML TAGS

```
pattern =  r"(\<(/?[^\>]+)\>)"


sentence = "<head> </head> <body  <div>"


result = re.findall(pattern, sentence)
```

ASSIGNMENT

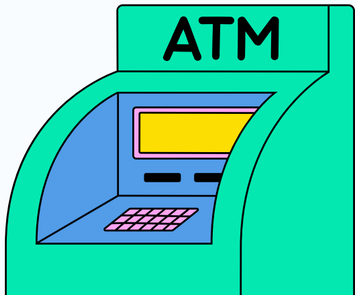
2

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

ATM SIMULATOR

Rules:

- Create ATM simulator using class and object in python
- It should check username and password and allow the user to use ATM
- It should have balance, withdrawl, and deposit options
- Everytime the user tries to use the options, send 6 digit OTP using random module and ask the user to enter that OTP and verify



THANKS FOR WATCHING

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED