بقالنا كام درس بنتعلم عن ال **Regular Expressions** ك concept لكن فى الدرس دا هنبتدي نطبق استخدامها في البايثون.

اول حاجه علشان اقدر استخدم ال **Regular Expressions** في البايثون بعمل import ل module اسمه re (**import** re) ودا اللى بيكون جواه ال methods اللي بتخليني اقدر اتعامل مع ال **Regular Expressions**.

اول method عندي وهي **re.search()** ودي بقدر اعمل بيها match على string معين واعرف هل بيطابق ال pattern بتاعي ولا لا. ال arguments بتاع الفانكشن بتاخد ال pattern بتاعي اللى عايز اطابق عليه و ال string اللى هدور فيه علي ال match ل pattern بتاعي.

ال re.search() بترجعلي اول match عندي في ال string بس، حتى لو في ال match في ال string تاني مش هترجع برضو غير اول واحد بس.

في ال output بتاع ال Search لو لقى match بيخرجهولي وبيخرجلي span() ودا بيكون ال indexes اللي واقع فيها ال pattern بتاعي (طبعا ال index الاخير عارفين انه not included).

طبعا اقدر اخرج ال span بس او حتى ال string اللى بدور فيه او كمان ال groups لو ال pattern بتاعي فيه groups.

ال method التانيه عندنا وهي **re.findall()** ودي بترجعلي كل ال matches اللي مطابقه ل pattern بتاعي فى ال string اللى بدور فيه. طبعا ال arguments بتوعها زي ال re.search().

ال output بتاعها بيكون list فاضيه لو ملقيتش اي match او بيكون list فيها كل ال matches اللي لقيتها في ال string.

في المثال الاخير بعمل list فاضيه وبطلب من ال user يدخل ايميل، وبعدين انا عامل pattern لشكل الايميل فبحط الاتنين واطابقهم ببعض. بعدين بعمل check وبقله لو ال list بتاع ال findall() مش فاضيه (يعني لقى match ودا معناه ان الايميل اللي دخله ال user مطابق ل pattern بتاعي) حطلي الايميل دا جوا ال list اللي انا كنت عاملها وفاضيه، ولو مش مطابق قله ان دا invalid Email.

```
# ------------------------------------------------------------
# -- Regular Expressions => Re Module Search And FindAll --
# ------------------------------------------------------------
# search()  => Search A String For A Match And Return A First Match Only
# findall() => Returns A List Of All Matches and Empty List if No Match
# ------------------------------------------------------------------------
# Email Pattern => [A-z0-9\.]+@[A-z0-9]+\.(com|net|org|info)
# ------------------------------------------------------------

import re

my_search = re.search("[A-Z]", "MuhammadAhmed")

my_search = re.search(r"[A-Z]{2}", "MMuhammadAAhmed")

print(my_search)
print(my_search.span())
```

```python
print(my_search.string)
print(my_search.group())

is_email = re.search(r"[A-z0-9\.]+@[A-z0-9]+\.(com|net)", "mu@abusenna.com")

if is_email:

    print("This is A Valid Email")

    print(is_email.span())
    print(is_email.string)
    print(is_email.group())

else:

    print("This is Not A Valid Email")

email_input = input("Please Write Your Email: ")

search = re.findall(r"[A-z0-9\.]+@[A-z0-9]+\.com|net", email_input)

empty_list = []

if search != []:

    empty_list.append(search)

    print("Email Added")

else:

    print("Invalid Email")

for email in empty_list:

    print(email)
```

CODE

```
PS C:\Users\Muhammad\Documents\Python Course>
/first.py"
<re.Match object; span=(0, 1), match='M'>
<re.Match object; span=(0, 2), match='MM'>
(0, 2)
MMuhammadAAhmed
MM
This is A Valid Email
(0, 15)
mu@abusenna.com
mu@abusenna.com
Please Write Your Email: mu@@abusenna.aspx
Invalid Email
```

OUTPUT: case wrong Email

```
PS C:\Users\Muhammad\Documents\Python Course>
/first.py"
<re.Match object; span=(0, 1), match='M'>
<re.Match object; span=(0, 2), match='MM'>
(0, 2)
MMuhammadAAhmed
MM
This is A Valid Email
(0, 15)
mu@abusenna.com
mu@abusenna.com
Please Write Your Email: abusenna44@gmail.com
Email Added
['abusenna44@gmail.com']
```

OUTPUT: case right Email