# Exercise: Regular Expressions

Problems for exercise and homework for the [Python Fundamentals Course @SoftUni](https://softuni.bg/trainings/3368/python-fundamentals-may-2021).  
Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1743>.

## Capture the Numbers

Write a program which receives **series of strings** **on different lines** and **extracts only the numbers**. Print all extracted numbers **on a single line**, separated by a **single space**.

import re  
  
text\_line\_list = []  
  
while True:  
 text\_line = input()  
 text\_line\_list.append(text\_line)  
 if not text\_line:  
 break  
  
text\_as\_string = ' '.join(text\_line\_list)  
  
pattern = r"\d+"  
valid\_nums = re.findall(pattern, text\_as\_string)  
  
print(\*valid\_nums)

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| The300  What is that?  I think it's the 3rd movie  Let's watch it at 22:45 | 300 3 22 45 |
| 123a456  789b987  654c321  0 | 123 456 789 987 654 321 0 |
| Let's go11!!!11!  Okey!1! | 11 11 1 |

## Find Variable Names in Sentences

Write a program which finds all **variable names** in a given string. A variable name starts with an **underscore** ("\_") and contains **only capital and non-capital letters and digits**. Extract only their names, **without the underscore.** Try to do this **only with regular expressions**.

The **output** consists of all variable names, **extracted,** and **printed on a single line**, **separated** by a **comma**.

import re  
  
text = input()  
  
pattern = r"(?<=\s)\\_[a-zA-Z]+\b"  
  
valid\_variable = re.findall(pattern, text)  
  
list\_valid\_variables = []  
  
for x in valid\_variable:  
 removed\_underscore = x[1:]  
 list\_valid\_variables.append(removed\_underscore)  
  
print(','.join(list\_valid\_variables))

Another solution using grouping in RegEx (?P<var\_name>) in order to print the word without the underscore:

import re  
  
text = input()  
  
pattern = r"(?<=\s)\\_(?P<var\_name>[a-zA-Z0-9]+)\b"  
  
valid\_variable = re.finditer(pattern, text)  
list\_valid\_variables = []  
  
for x in valid\_variable:  
 list\_valid\_variables.append(x.group('var\_name'))  
  
print(','.join(list\_valid\_variables))

|  |  |
| --- | --- |
| **Input** | **Output** |
| The \_id and \_age variables are both integers. | id,age |
| Calculate the \_area of the \_perfectRectangle object. | area,perfectRectangle |
| \_\_invalidVariable \_evenMoreInvalidVariable\_ \_validVariable | validVariable |

## Find Occurrences of Word in Sentence

Write a program which finds **how many times** a **given word** is **used** in a **given sentence.** Note that letter case does not matter – it is **case-insensitive**. The **output** is a single number indicating the **number of times** the sentence contains the word.

import re  
  
text = input().lower()  
search\_word = input().lower()  
words\_list = []  
search\_word\_list = []  
  
pattern = rf"\b{search\_word}\b"  
words = re.finditer(pattern, text)  
  
for x in words:  
 words\_list.append(x.group())  
  
for y in words\_list:  
 if y == search\_word:  
 search\_word\_list.append(search\_word)  
  
print(len(search\_word\_list))

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| The waterfall was so high, that the child couldn't see its peak.  the | 2 |
| How do you plan on achieving that? How? How can you even think of that?  how | 3 |
| There was one. Therefore I bought it. I wouldn't buy it otherwise.  there | 1 |

## Extract Emails

Write a program which receives a **single string** and **extracts all email addresses from it**. **Print** the extracted email addresses **on separate lines**. Emails are in the format "{user}@{host}", where:

* **{user}** could consist only of **letters** and **digits**; the symbols **"."**, **"-"** and **"\_"** can appear **between** them.
  + Examples of valid users: "**stephan**", **"mike03"**, **"s.johnson"**, **"st\_steward"**, **"softuni-bulgaria"**, **"12345"**
  + Examples of invalid users: **''--123"**, **"....."**, **"nakov\_-"**, **"\_steve"**, **".info**"
* **{host}** is a sequence of **at least two words**, each couple of words must be separated by a **single dot** **"."**. Each word consists of **only letters** and can have **hyphens** **"-"** **between** the letters.
  + Examples of valid hosts: **"softuni.bg"**, **"software-university.com"**, **"intoprogramming.info"**, **"mail.softuni.org"**
  + Examples of invalid hosts: **"helloworld"**, **".unknown.soft."** , **"invalid-host-"**, **"invalid-"**

Examples of valid emails: info@softuni-bulgaria.org, kiki@hotmail.co.uk, no-reply@github.com, s.peterson@mail.uu.net, info-bg@software-university.software.academy

Examples of invalid emails: **--123@gmail.com, …@mail.bg, .info@info.info, \_steve@yahoo.cn, mike@helloworld, mike@.unknown.soft.,** [**s.johnson@invalid-**](mailto:s.johnson@invalid-)

import re  
  
text = input()  
  
valid\_emails\_list = []  
  
pattern = r"(^|(?<=\s))(([a-zA-Z0-9]+)([\.\-\_]?)([A-Za-z0-9]+)(@)([a-zA-Z]+([\.\-\_][A-Za-z]+)+))(\b|(?=\s))"  
# 1. (^|(?<=\s)) - check if there's a start of a string OR ( " | ") if there's space (what the first group does).  
# 2. ([a-zA-Z0-9]+)([\.\-\_]?)([A-Za-z0-9]+)(@) - check : (word which contains letters and numbers);   
# (then check some of the allowed symbols IF there is, hence we use "?");  
# (then again word which contains letters and numbers); the words number can be 1.  
# 3. (include the symbol @)  
# 4. ([a-zA-Z]+([\.\-\_][A-Za-z]+)+) - search for word;  
# but after the word there SHOULD be any of the allowed symbols ".", "-" or "\_";  
# otherwise the string will be faulty with having only one word;  
# because the problem says there should be at least 2 words;  
# finally the domain - a (.bg, .net and so on), for example @mail.uu.net;   
# ([a-zA-Z]+([\.\-\_][A-Za-z]+)+) - the plus sign is crucial for group 8,  
# as it seeks two more matches ... you can mark more words, but they SHOULD  
# start with any of the allowed symbols.  
# 5. (\b|(?=\s)) the same as 1. - checks for boundary \b or positive lookahead   
# to check if there is a white space.  
  
match = re.finditer(pattern, text)  
  
for x in match:  
 valid\_emails\_list.append(x.group())  
  
# print(valid\_emails\_list)  
print('\n'.join(valid\_emails\_list))

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Please contact us at: support@github.com. | support@github.com |
| Just send email to s.miller@mit.edu and j.hopking@york.ac.uk for more information. | s.miller@mit.edu  j.hopking@york.ac.uk |
| Many users @ SoftUni confuse email addresses. We @ Softuni.BG provide high-quality training @ home or @ class. –- steve.parker@softuni.de. | steve.parker@softuni.de |

## Furniture

Write a program which calculates **the total cost** of bought furniture. You will be given **information about each purchase on separate lines** until you receive the command **"Purchase"**. **Valid information** should be in the format: **">>{furniture\_name}<<{price}!{quantity}"**. The **price** could be **floating-point** **or** **integer** **number**. You should store the names of the furniture and the total price.

At the end, print the **name of each bought furniture** and **the total cost**, formatted to the **second decimal point**:

**"Bought furniture:**

**{1st name}**

**{2nd name}**

**…**

**{N name}**

**Total money spend: {total\_cost}"**

import re  
  
command = input()  
purchase\_list = []  
total\_cost = 0  
  
while command != "Purchase":  
 purchase\_list.append(command)  
  
 command = input()  
  
pattern = r">>(?P<furniture>[A-Za-z]+)<<(?P<price>\d+(.\d+)?)!(?P<qty>\d+)"  
  
print("Bought furniture:")  
  
for x in purchase\_list:  
 match = re.finditer(pattern, x)  
 for y in match:  
 total\_cost += float(y.group('price')) \* int(y.group('qty'))  
 print(y.group('furniture'))  
  
print(f"Total money spend: {total\_cost:.2f}")

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| >>Sofa<<312.23!3  >>TV<<300!5  >Invalid<<!5  Purchase | Bought furniture:  Sofa  TV  Total money spend: 2436.69 | Only the Sofa and the TV are valid, for each of them we multiply the price by the quantity and print the result |

## \*Extract the Links

Write a program which **extracts links** from a **given text**. The text will come in the form of strings, each representing a sentence. You need to extract **only the valid links** from it. Example:

"www.internet.com"

**Sub-Domain**  **Domain** **name** **Domain extension**

The **Sub-Domain** must always be "www". The **Domain name** can consist of English alphabet letters (**uppercase** and **lowercase**), digits and dashes (**"**–**"**). The **Domain extension** consists of one or more **domain blocks**, a **domain block** consists of a **dot** followed by **one or more lowercase** English alphabet **letters**, a **Domain extension** must have at least **one** **domain block** in order to be **valid**. The Sub-Domain and Domain name must be separated by a single **dot**. Any link that **does NOT follow** the specified above rules should be treated as **invalid**.

**Example incorrect links:**

* **"ww.justASite.bg"**
* **"lel.awesome.com"**
* **"www.weird\_site.hit.bg"**
* **"www.no-symb#^ols-allow%ed.com"**
* **"www.mark.12"**
* **"www.web-site."**
* **"www.example-site.\_\*^#"**

**Example of correct links:**

* **"Some textwww.softuni.bg"**
* **"Just a link in a www.sea-of-text.bg-swimming around"**
* **"Instruments www.Instruments.rom.com.trombone2000 Instrument here"**
* **"All your ice cream flavors-www.scream.for.ice.cream(We also have squirrels)"**

The **output** is all valid links you have found, printed – each on a new line.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Join WebStars now for free, at [www.web-stars.com](http://www.web-stars.com/)  You can also support our partners:  Internet - [www.internet.com](http://www.internet.com/)  WebSpiders - [www.webspiders101.com](http://www.webspiders101.com/)  Sentinel - www.sentinel.-ko | www.web-stars.com  www.internet.com  www.webspiders101.com |
| Need information about cheap hotels in London?  You can check us at [www.london-hotels.co.uk](http://www.london-hotels.co.uk) !  We provide the best services in London.  Here are some reviews in some blogs:  London Hotels are awesome! - [www.indigo.bloggers.com](http://www.indigo.bloggers.com/)  I am very satisfied with their services - ww.ivan.bg  Best Hotel Services! - [www.rebel21.sedecrem.moc](http://www.rebel21.sedecrem.moc/) | www.london-hotels.co.uk  www.indigo.bloggers.com  www.rebel21.sedecrem.moc |

import re  
  
text = input()  
text\_list = []  
  
while text:  
 text\_list.append(text)  
 text = input()  
  
pattern = r"(www)\.([A-Za-z][A-Za-z0-9-]+)(\.[a-z]+)+"  
  
for x in text\_list:  
 match = re.finditer(pattern, x)  
 for y in match:  
 print(y.group())