## Match Phone Number

Write a regular expression to match a **valid phone number** from **Sofia**. After you find all **valid phones**, **print** them on the console, separated by a **comma and a space** “, ”.

### Compose the Regular Expression

A valid number has the following characteristics:

* It starts with "**+359**"
* Then, it is followed by the area code (always **2**)
* After that, it’s followed by the **number** itself:
  + The number consists of **7 digits** (separated in **two** **groups** of **3** and **4** **digits** respectively).
* The different **parts** are **separated** by **either a space or a hyphen** ('**-**').

You can use the following table of values to test your RegEx against:

|  |  |
| --- | --- |
| **Match ALL of these** | **Match NONE of these** |
| +359 2 222 2222  +359-2-222-2222 | 359-2-222-2222, +359/2/222/2222, +359-2 222 2222  +359 2-222-2222, +359-2-222-222, +359-2-222-22222 |

## Match Hexadecimal Numbers

Write a program, which finds all **valid hexadecimal numbers** in a **string** and **print** them **space-separated**.

### Compose the Regular Expression

A valid hexadecimal number follows these conditions:

* Can have “0x” in front of it (not required)
* Has **one or more** **hexadecimal** **digits** after it (0-9 and A-F).
* Doesn’t have anything on **either** of its sides (use \b).

You can follow the table below to help with composing your RegEx:

|  |  |
| --- | --- |
| **Match ALL of these** | **Match NONE of these** |
| 0x10 0xAB 0x1F 10 AB 1F FF | 0xG G 0x4G 4G 0xFG FG |

Find all the **hexadecimal numbers** from the string and **print them** on the **console**, separated by **spaces**.

## Match Dates

Write a program, which matches a date in the format “dd{separator}MMM{separator}yyyy”. Use **named** **capturing groups** in your regular expression.

### Compose the Regular Expression

Every valid date has the following characteristics:

* Always starts with **two digits**, followed by a **separator**
* After that, it has **one uppercase** and **two lowercase** letters (e.g. Jan, Mar).
* After that, it has a **separator** and **exactly 4 digits** (for the year).
* The separator could be either of three things: a period (“.”), a hyphen (“-“) or a forward slash (“/”)
* The separator needs to be **the same** for the whole date (e.g. 13.03.2016 is valid, 13.03/2016 is **NOT**). Use a **group backreference** to check for this.

You can follow the table below to help with composing your RegEx:

|  |  |
| --- | --- |
| **Match ALL of these** | **Match NONE of these** |
| 13/Jul/1928, 10-Nov-1934, 25.Dec.1937 | 01/Jan-1951, 23/sept/1973, 1/Feb/2016 |

## Extract emails

Write a program to **extract all email addresses from a given text**. The text comes at the only input line. Print the emails on the console, each at a separate line. Emails are considered to be in format **<user>@<host>**, where:

* **<user>** is a sequence of letters and digits, where '**.**', '**-**' 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, separated by dots '**.**'. Each word is sequence of letters and can have hyphens '**-**' between the letters. Examples of 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](mailto:info-bg@software-university.software.academy).
* Examples of **invalid emails**: [--123@gmail.com](mailto:--123@gmail.com), …@mail.bg, [.info@info.info](mailto:.info@info.info), [\_steve@yahoo.cn](mailto:_steve@yahoo.cn), mike@helloworld, [mike@.unknown.soft](mailto:mike@.unknown.soft)., [s.johnson@invalid-](mailto:s.johnson@invalid-).

### 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* |

## Valid Usernames

You are part of the back-end development team of the next Facebook. You are given **a line of usernames**, between one of the following symbols**: space, “/”, “\”, “(“, “)”.** First you have to export all **valid** usernames. A valid username **starts with a letter** and can contain **only letters, digits and “\_”**. It cannot be **less than 3 or more than 25 symbols** long. Your task is to **sum** the length of **every** **2 consecutive** **valid** usernames and print on the console the 2 valid usernames with **biggest** **sum** of their **lengths,** each on a separate line.

### Input

The input comes from the console. One line will hold all the data. It will hold **usernames,** divided by the symbols: **space, “/”, “\”, “(“, “)”.**

The input data will always be valid and in the format described. There is no need to check it explicitly.

### Output

Print at the console the 2 **consecutive** **valid usernames** with the **biggest** **sum** of their lengths each on a separate line. If there are **2 or more couples** of usernames with the same sum of their lengths, print he **left most**.

### Constraints

* The input line will hold characters in the range [0 … 9999].
* The usernames should **start with a letter** and can contain **only letters, digits and “\_”**.
* The username cannot be **less than 3 or** **more than 25 symbols** long.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| ds3bhj y1ter/wfsdg 1nh\_jgf ds2c\_vbg\4htref | wfsdg  ds2c\_vbg |

|  |  |
| --- | --- |
| **Input** | **Output** |
| min23/ace hahah21 ( sasa ) att3454/a/a2/abc | hahah21  sasa |

|  |  |
| --- | --- |
| **Input** | **Output** |
| chico/ gosho \ sapunerka (3sas) mazut lelQ\_Van4e | mazut  lelQ\_Van4e |