RegExp Master

You are given a set of 10 tasks. For each of these tasks you must write the correct Regular Expression of minimal size in C format (current SPOJ version). Each regular expression will be tested against a special test set, which contains right and wrong strings. The expression which is composed by you must correctly work on all test cases.

Task 1 You are to write regular expression which determines if a given string is equal to "abcdefghijklmnopqrstuv18340" or not.

Example of correct strings:
abcdefghijklmnopqrstuv18340
Example of wrong strings:
abcdefghijklmnoasdfasdpqrstuv18340

Task 2 You are to write a regular expression which determines whether a given string is a GUID, with or without brackets. Here GUID is a string, consisting of 8, 4, 4, 4, 12 hex digits separated by '-'.

Examples of correct strings:
{e02fa0e4-01ad-090A-c130-0d05a0008ba0}
e02fd0e4-00fd-090A-ca30-0d00a0038ba0

Examples of wrong strings:
02fa0e4-01ad-090A-c130-0d05a0008ba0}
e02fd0e400fd090Aca300d00a0038ba0

Task 3 You are to write a regular expression which determines whether the given string is a valid MAC-address.

Examples of correct strings:
01:32:54:67:89:AB
aE:dC:cA:56:76:54

Examples of wrong strings:
01:33:47:65:89:ab:cd
01:23:45:67:89:Az

Task 4 You are to write a regular expression which determines whether a given string is uppercase and sorted in non-descending order.

Examples of correct strings:
AABCD
ABCDZ

Examples of wrong strings:
aABCD
ZABCD

Task 5 You are to write a regular expression which determines whether a given string is the hex

identification of a color in HTML. Here #FFFFFF stands for white, #000000 for black, #FF0000 for red, etc.

Examples of correct strings:
#FFFFFF
#FF3421
#00ff00

Examples of incorrect strings:
232323
f#fddee
#fd2

Task 6 You are to write a regular expression which determines whether the given string is a date in dd/mm/yyyy format. The date is in the range from the year 1600 to the year 9999.

Examples of correct strings:
29/02/2000
30/04/2003
01/01/2003

Examples of wrong strings:
29/02/2001
30-04-2003
1/1/1899

Task 7 You are to write a regular expression which determines whether the given string is a valid e-mail address with respect to RFC number 2822

Examples of correct strings:
mail@mail.ru
valid@megapochta.com
aa@aa.info

Examples of wrong strings:
bug@@@com.ru
@val.ru
Just Text2
val@val
val@val.a.a.a.a
12323123@111[]][]

Task 8 You are to write a regular expression which determines whether the given string is an IP address, in decimal format

Examples of correct strings:
127.0.0.1
255.255.255.0
192.168.0.1

Examples of wrong strings:
1300.6.7.8
abc.def.gha.bcd

254.hzf.bar.10

Task 9 You are to check whether a given password is strong. A password is said to be strong if it consists of 8 and more symbols, where a symbol is one from the set: English letter, digit or underline. Additionally, a strong password must contain at least one uppercase letter, at least one lowercase letter and at least one digit.

Examples of correct strings:
C00l_Pass
SupperPas1

Examples of wrong strings:
Cool_pass
C00l

Task 10 You are to write a regular expression which determines whether a given string is a six-digit positive integer, printed in decimal format without leading zeros.

Examples of correct strings:
123456
234567

Examples of wrong strings:
1234567
12345

For testing we use the following C-function:

```
int match(const char *string, char *pattern)
{
    int status;
    regex_t re;

    if (regcomp(&re, pattern, REG_EXTENDED|REG_NOSUB) != 0) {
        return(0);
    }
    status = regexec(&re, string, (size_t) 0, NULL, 0);
    regfree(&re);
    if (status != 0) {
        return (0);
    }
    return (1);
}
```

Input

There is no input data for this problem

Output

Output your answer as a set of 10 lines. The first line is for the first task, the second line for the second task, etc. All other lines will be ignored. If you don't want to solve some task, then in the corresponding line output "---". Otherwise, output the regular expression for this task. If any of

your regular expressions are invalid you'll get Wrong Answer status.

Score

For each solved task you'll get exactly 1 point plus a bonus points equal to 1/(regular expression size).

Example

Output:

```
---
^[1-9]{1}[0-9]{3} ?[A-Z]{2}$
---
---
---
---
---
---
---
---
```

It's just an example of what output data should look like. If the answer for second task were right, then you would get 1 + 1/28 = 1.035714 points.