

Day 28: RegEx, Patterns, and Intro to Databases!

Welcome to Day 28! Check out an [Introduction to Databases](#), or jump into the challenge. We haven't discussed *RegEx*(Regular Expressions) yet, but that's okay! Review the [Pattern documentation](#), learn what it can do, and apply your new knowledge to this challenge!

RegEx helps us easily search for or match a *Pattern* in text. Before searching for a *Pattern*, we must specify it using some well-defined syntax.

Given a string, determine if it's a valid *Pattern* or not. The string may contain spaces.

Note: This is a java only challenge, a *RegEx* is only valid if you can *compile* it using the [Pattern.compile](#) method. You may find using a *try-catch* block helpful here.

Input Format

The first line of input contains an integer, (the number of test cases).

The subsequent lines of test cases each contain a string of characters describing a *RegEx*.

Constraints

Output Format

On a new line for each test case, print **Valid** if the given *RegEx*'s syntax is correct; otherwise, print **Invalid**.

Sample Input

```
3
([A-Z])(.+)
[AZ[a-z](a-z)
batcatpat(nat
```

Sample Output

```
Valid
Invalid
Invalid
```

Explanation

The second and third test cases have unbalanced brackets and will throw a [PatternSyntaxException](#) when compiled. For example:

`[AZa-z` is **Invalid**, but `[AZa-z]` would be **Valid**.

`batcatpat(nat` is **Invalid**, but `batcatpat(nat)` would be **Valid**.