Problem 252: Lost in Translation

Difficulty: Easy

Author: Dr. Francis Manning, Syracuse, New York, United States

Originally Published: Code Quest 2025

Problem Background

When working with embedded control systems, it's often necessary to use hexadecimal notation to send command strings to the device to instruct it to perform a given action. This helps reduce the complexity of the validation code needed to ensure the safe operation of the device.

Problem Description

You'll be provided with a series of strings containing hexadecimal values that represent command signals sent to an embedded device. You'll need to translate these values back into recognizable characters, then check to see if all of those characters are valid.

For example, a string such as...

```
Reset motor control
```

...might have been encoded into hexadecimal as...

```
52 65 73 65 74 20 6D 6F 74 6F 72 20 63 6F 6E 74 72 6F 6C
```

Each character is encoded using its ASCII value; 'R', for example, has a decimal value of 82 in ASCII. When converted to hexadecimal, it is equal to 52.

Once you decode the message back to its original form, you'll need to check to ensure all of the characters are recognizable by the embedded system. The valid characters include:

- All letters (uppercase and lowercase)
- All numbers (0-9)
- Spaces
- The following punctuation marks:
    - Period (.)
    - Comma (,)
    - Square brackets ([and])
    - Colons (:)

The hexadecimal strings may include encoded invalid characters; this may occur if the data is corrupted or was sent by a malicious attacker. Such strings must be identified so we can protect the embedded systems.

Sample Input

The first line of your program's input, **received from the standard input channel**, will contain a positive integer representing the number of test cases. Each test case will include a single line containing a series of pairs of uppercase hexadecimal digits, with each pair separated by a space. These values represent the encoded command strings.

```
2
52 65 73 65 74 20 6D 6F 74 6F 72 20 63 6F 6E 74 72 6F 6C
43 4F 44 45 20 71 75 65 24 74 21
```

Sample Output

For each test case, your program must print a single line with the word 'VALID' if all characters in the provided string are valid, or the word 'INVALID' otherwise.

```
VALID
INVALID
```