'Sed' command #4

HackerRank

Sed is a popular utility that enables quick parsing and transformation of text. Here are some basic uses for it:

Substitute the *first* occurrence of `editor` with `tool`:

```
$:~/hackerrank/bash/grep/grep1$ echo "My favorite programming editor is Emacs. Another editor I like is Vim."
| sed -e s/editor/tool/
My favorite programming tool is Emacs. Another editor I like is Vim.
```

Substitute all occurrences of `editor` with `tool`:

```
$:~/hackerrank/bash/grep/grep1$ echo "My favorite programming editor is Emacs. Another editor I like is Vim."
| sed -e s/editor/tool/g
My favorite programming tool is Emacs. Another tool I like is Vim.
```

Substitute the *second* occurrence of `editor` with `tool`:

```
$:~/hackerrank/bash/grep/grep1$ echo "My favorite programming editor is Emacs. Another editor I like is Vim."
| sed -e s/editor/tool/2
My favorite programming editor is Emacs. Another tool I like is Vim.
```

Highlight all occurrences of `editor` by enclosing them in curly brackets (i.e., `{}`):

```
$:~/hackerrank/bash/grep/grep1$ echo "My favorite programming editor is Emacs. Another editor I like is Vim."
| sed -e s/editor/{\&}/g
My favorite programming {editor} is Emacs. Another {editor} I like is Vim.
```

Task

Given n lines of credit card numbers, mask the first 12 digits of each credit card number with an asterisk (i.e., `*`) and print the masked card number on a new line. Each credit card number consists of four space-separated groups of four digits. For example, the credit card number `1234 5678 9101 1234` would be masked and printed as `**** **** **** 1234`.

References

You may find the following links helpful in learning about `sed`:

- Sed: An Introduction and Tutorial
- The TLDP Guide
- Some Practical Examples
- A StackOverflow question on a slightly modified version of this task where the solution involves backreferences.
- A ttuorial from TheGeekStuff detailing the use of groups and backreferences.

Input Format

Each line contains a credit card number in the form `dddd dddd dddd dddd`, where d denotes a decimal digit (i.e., 0 through 9). There are a total of n lines of credit card numbers.

Constraints

- $1 \leq n \leq 20$; note that the value of n does not matter when writing your command.

Output Format

For each credit card number, print its masked version on a new line.

Sample Input

```
1234 5678 9101 1234
2999 5178 9101 2234
9999 5628 9201 1232
8888 3678 9101 1232
```

Sample Output

```
**** **** **** 1234
**** **** **** 2234
**** **** **** 1232
**** **** **** 1232
```

Explanation

Observe that the first twelve digits have been masked for each credit card number, and they are printed in the same order as they were received as input.