Problem 120: Hidden In Plain Sight

Difficulty: Medium

Author: Holly Norton, Fort Worth, Texas, United States

Originally Published: Code Quest Australia 2019

Problem Background

Sending secret messages isn't all about codes and ciphers. Sometimes you can send a secret message by hiding it in plain sight. This practice, called *steganography*, involves hiding a message within an otherwise innocent-seeming different message, so that nobody realizes the hidden message exists at all.

Steganography has taken on many different forms throughout history. The oldest known use of it dates back to ancient Greece. At the time, wax tablets held in wooden frames were commonly used for writing, since they could be easily erased (melted) and reused. Demaratus, a Greek exile living in Persia, became aware of the Persians' plans to attack Greece. He melted the wax out of a tablet, engraved a warning on the wooden frame beneath, then replaced the wax and wrote an innocuous message on top. When the message arrived in Sparta, the Spartans discovered the hidden message the Persians had missed and were able to prepare for the attack.

Today, messages can be hidden in any number of ways. Images are a particular favorite; slightly changing the color of certain pixels in the image or placing extra information in an image's metadata allows anyone to send a message through something as simple as a meme without letting on that there's more than meets the eye. Messages can also be hidden in websites, emails, video feeds, sound clips... in a way, computers have made steganography easier than ever before.

Problem Description

Your program should find a hidden message in seemingly random words by using the given (zero-based) character index for each line. The characters from each line should be put together to form the secret message.

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 consist of the following input:

- A positive integer, X, representing the number of lines to read in the test case.
- X lines containing a text string ending with a pipe (|) and a non-negative integer, N. The text string may include any printable character other than a pipe.

```
1
14
Earth|4
Breakfast|2
lamp|0
family|4
cookie|1
Dear, Bob|4
Six Flags|3
Candy|0
coffee|1
on the edge|8
the|2
America|3
Split-second|6
Wow!|3
```

Sample Output

For each test case, your program must print a single line that is X characters long, consisting of the Nth character of each text string. Retain the original case for each character.

```
hello, Coders!
```