Contest IV: 1997 Leuven, Belgium

1. Snake

The body of the Belgian snake shows a repeating pattern. However, the pattern is not necessarily repeated an integral number of times. A pattern consists of a sequence of rings, and a ring has an identifier which is an atom of length 1. When the Belgian snake takes a nap, it likes to lie coiled up in a particular way: it always lies in a rectangle, its head in the upper left corner and filling the rectangle row by row (see the query below). Write a predicate `snake/3`, which displays such a coiled up Belgian snake. This predicate will be called with the following three arguments: a list of atoms representing a pattern, a list whose length is equal to the number of rings in one column and a list whose length is equal to the number of rings in one row. Your `snake/3` should draw the coiled up snake as output on the screen. For example:

```
?- snake([a,b,c,d],[_,_,_,_,_],[_,_,_]).

abcda
badcb
cdabc
```

This snake would look like `abcdabcdabcdabc` when stretched out.

There is a catch: Belgian snakes dislike arithmetic computations very much. Therefore, we urge you to avoid any arithmetic.

Hints *The snake consists of an ever repeating pattern, and one way of representing this is by a cyclic list: it contain the pattern and bites its tail. This cyclic list is used as a infinite supply of the pattern, from which we need to take pieces with the same length as the list in the second argument. This piece must be reversed for even rows.*
