Grammars P45126_en

A grammar consists of rules composed by one or more productions and one or more terminal symbols. In this exercise we suppose that the productions start with an uppercase letter, and that the initial production is called I. For instance, the following grammar generates all the (non empty) correct parenthesizations:

$I \rightarrow (\ ) \mid (\ I \ ) \mid I \ I$

In this example, **( )** is generated by applying the first rule, (( )) is generated by applying the second rule and then the first one, ( ) ( ) is generated by applying the third rule and then the first one twice, etcetera.

Assume that *lambda* denotes an empty special terminal symbol. Using this convention, the following grammar generates all the words that start with *a* and that contain an even number of *a*s and an odd number of *b*s:

$$
\begin{array}{lcll}
I & \rightarrow & a \ OO & \\
OO & \rightarrow & a \ EO & \mid \ b \ OE \\
OE & \rightarrow & a \ EE & \mid \ b \ OO \\
EO & \rightarrow & a \ OO & \mid \ b \ EE \\
EE & \rightarrow & lambda & \mid \ a \ OE \ \mid \ b \ EO
\end{array}
$$

Write a program that prints n terms generable with a given grammar. If a production has r rules (numbered from 0 to $r-1$), to decide which rule to pick, use a generator of pseudorandom numbers with parameters m, a, b and s, as it is explained in the exercise P33549: "Random paths":

```
int random(int r, int m, int a, int b, int& s) {
    s = (a*s + b)%m;
    return s%r;
}
```

Input

Input starts with a natural number $p > 0$ that indicates the number of productions. Each production starts with its name and its number of rules $r > 0$. Each rule is described with a natural number $m > 0$ followed by m names of productions or terminal symbols.

Input ends with the number n of terms that must be generated, followed by the four natural numbers m, a, b and s that define the generator of pseudorandom numbers.

Output

Print n lines, each one with the term that results after applying the rules indicated by the generator of pseudorandom numbers. Use the productions from left to right inside every rule. Each word must be preceded by a space.

Observation

The first sentence of the third sample output is the literal translation of a Catalan tongue twister.

Sample input 1

```
1
I 3
  2 ( )
  3 ( I )
  2 I I
8
1007 499 7 400
```

Sample output 1

```
 ( ( ) ) ( ( ) )
 ( ) ( )
 ( )
 ( )
 ( ( ) )
 ( )
 ( ( ( ) ) ( ) )
 ( ( ) )
```

Sample input 2

```
5
I 1
  2 a OO
OO 2
  2 a EO
  2 b OE
OE 2
  2 a EE
  2 b OO
EO 2
  2 a OO
  2 b EE
EE 3
  1 lambda
  2 a OE
  2 b EO
15
987 297 15 299
```

Sample output 2

```
 a b a
 a a a b b b b b a
 a a a a a a a b a
 a a a b b b b b b b a
 a a a a b
 a a b
 a b b b b b a
 a b b a a b a
 a b b a b
 a a a a b
 a a a b a
 a b a
 a a b
 a b a
 a b b b a
```

Sample input 3

```
6
Obj 2
  1 Sin
  1 Plu
Sin 2
  1 Y
  3 Y of Obj
Plu 2
  1 Z
  3 Z of Obj
Y 3
  2 a court
  1 liver
  3 a hanged man
Z 1
  2 sixteen judges
I 2
  3 Plu eat Obj
  3 Sin eats Obj
5
100207 4517 9843 29253
```

Sample output 3

```
sixteen judges of a court eat liver of a hanged man
a hanged man eats a hanged man
a court of liver eats liver of a hanged man
a court eats sixteen judges
sixteen judges of sixteen judges eat a court
```

Problem information

Author : Salvador Roura
Translator : Carlos Molina
Generation : 2013-09-02 14:50:48