

Round C APAC Test 2017

[A. Monster Path](#)

[B. Safe Squares](#)

C. Evaluation

[D. Soldiers](#)

[Questions asked](#)

Submissions

Monster Path

7pt	Not attempted 752/1194 users correct (63%)
8pt	Not attempted 655/740 users correct (89%)

Safe Squares

6pt	Not attempted 1460/1651 users correct (88%)
13pt	Not attempted 621/1296 users correct (48%)

Evaluation

12pt	Not attempted 625/943 users correct (66%)
15pt	Not attempted 552/615 users correct (90%)

Soldiers

16pt	Not attempted 106/239 users correct (44%)
23pt	Not attempted 24/63 users correct (38%)

Top Scores

johngs	100
NAFIS	100
nathanajah	100
asdsteven	100
hello92world	100
pkwv	100
Sumeet.Varma	100
akulsareen	100
nhho	100
aguss787	100

Problem C. Evaluation

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the [Quick-Start Guide](#) to get started.

Small input
12 points

Solve C-small

Large input
15 points

Solve C-large

Problem

Given an unordered list of assignment statements, write a program to determine whether the assignment statements can be put in some order in which all variables can be evaluated.

For our problem, an assignment statement will consist of an assignment variable, an assignment operator, and an expression, in that order. Statements will be evaluated one at a time, in the order you choose for them. A variable can be evaluated if and only if it has been the assignment variable of a previous assignment statement.

To simplify the problem, all the expressions are single function calls. Functions can take an arbitrary number of arguments, including zero; a function with zero arguments is always valid, and a function with variable arguments is valid as long as all of the variables are evaluable.

For example, for the following list of assignment statements:

```
a=f(b,c)
b=g()
c=h()
```

this is one order that makes every statement valid:

```
b=g()
c=h()
a=f(b,c)
```

This is because: (1) `b` and `c` can be evaluated because the expressions `g()` and `h()` don't depend on any variables; and (2) `a` can also be evaluated because expression `a` depends on `b` and `c`, which are evaluable.

However, the order

```
b=g()
a=f(b,c)
c=h()
```

would not be valid, because `f(b, c)` has variable `c` as an argument, but variable `c` has not been an assignment variable yet.

Another example is: `a=f(a)`. This list of statements can't be evaluated because the expression `f(a)` depends on the variable `a` itself, which makes it impossible to evaluate the statement.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. The first line of each test case contains an integer **N**: the number of assignment statements. Then, **N** lines follow. Each contains one assignment statement.

Each assignment statement consists of three parts: the assignment variable, the assignment operator, and the expression, with no spaces in between. The assignment operator is always `=`. All expressions consist of a function name, then `(`, then zero or more comma-separated variable names, then `)`. All variables and function names consist of one or more lowercase English alphabet letters. No variable has the same name as a function. No variable will appear more than once as the assignment variable. However, variables may appear more than once in various functions (even within the same function), and functions may appear more than once.

Output

For each test case, output one line containing Case `#x`: `y`, where `x` is the test case number (starting from 1) and `y` is `GOOD` if all variables are evaluable or `BAD` otherwise.

Limits

$1 \leq T \leq 20$.

All functions take between 0 and 10 arguments, inclusive. All variable names consist of between 1 and 20 lowercase English alphabet letters.

Small dataset

$1 \leq N \leq 100$.

Large dataset

$1 \leq N \leq 1000$.

Sample

Input	Output
4	Case #1: GOOD
3	Case #2: BAD
a=f(b,c)	Case #3: BAD
b=g()	Case #4: GOOD
c=h()	
2	
a=f(b)	
b=f(a)	
2	
aaa=foo(x,y)	
bbb=bar(aaa,bbb)	
2	
x=f()	
y=g(x,x)	

All problem statements, input data and contest analyses are licensed under the [Creative Commons Attribution License](#).

© 2008-2017 Google [Google Home](#) - [Terms and Conditions](#) - [Privacy Policies and Principles](#)

Powered by



Google Cloud Platform