**ЗАДАЧА A1. ИЗРАЗИ**

**Автор: Павлин Пеев**

|  |  |  |
| --- | --- | --- |
| *x* | *y* | *xy* |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Зададена е функция с два аргумента, чиито стойности могат да са само целите числа 0 и 1. Тъй като това е единствената функция, която ще разглеждаме, няма да използваме някакъв символ за означаването ѝ. Резултатите от прилагането на тази функция върху възможните двойки аргументи също са от множеството {0,1} и са зададени в таблицата.

Ще изчисляваме изрази, които съдържат само променливи, върху които, евентуално, е приложена само тази функция. Променливите ще са означени с малки латински букви. При това, ако в израза има *n* променливи, те ще бъдат означавани точно с първите *n* латински букви. Така изразите, които ще разглеждаме, ще изглеждат като символни редици, съставени от първите няколко малки латински букви и, евентуално, правилно разположени малки скоби. Ето и формална дефиниция на подобни изрази:

* всяка малка латинска буква е *израз*:

<**expr**>::=*a*|*b*|*c*|*d*|*e*|*f*|*g*|*h*|*i*|*j*|*k*|*l*|*m*|*n*|*o*|*p*|*q*|*r*|*s*|*t*|*u*|*v*|*w*|*x*|*y*|*z*

* *израз* в скоби е *израз*:

<**expr**>::=(<**expr**>)

* *израз*, записан до друг *израз*, е *израз* (и между двата се предполага дефинираната функция):

<**expr**>::=<**expr**><**expr**>

Разликата е само, че, според формалната дефиниция, *ac*, например, е израз, но заради зададеното допълнително ограничение (че ще се срещат точно първите букви от азбуката), *c* ще бъде „преименувано“ в *b* и такъв израз ще бъде „еквивалентно“ зададен като *ab*.

При липса на скоби, изчисленията се извършват *от ляво надясно*, т.е., изразът *abc* ще се изчислява като (*ab*)c.

Напишете програма **expr**, която изчислява стойности на даден израз за зададени набори от стойности на променливите, които участват в него.

**Вход**

Първият ред на стандартния вход съдържа низ, който представлява коректен израз според горната дефиниция.

Всеки от следващите редове, освен последният, задава набор от стойности за променливите, срещащи се в израза от първия ред, т.е. той съдържа само символи 0 и 1, общо толкова, колкото е броят на различните променливи в израза, зададен на първия входен ред. Първият символ от всеки ред задава стойността на променливата *a*, вторият (ако има такъв) – стойността на променливата *b* и т. н. Входът завършва с ред, единственият символ в който е 2.

**Изход**

Програмата трябва да изведе на стандартния изход един ред, който съдържа само символите 0 и 1. Това са резултатите от изчисляването на израза за наборите от стойности на променливите, зададени в редовете от входа, в същата последователност.

**Ограничения**

Входът съдържа поне три и не повече от 200 002 реда. Описаният в първия ред израз не е по-дълъг от 1000 символа. Във входа няма два реда, които задават един и същ набор от стойности за променливите.

В 20% от тестовите примери броят на променливите не надвишава 5.

**Пример**

**Вход**

a(a)(ab(ca)bb)c

011

100

010

111

001

2

**Изход**

01101

***Обяснение към примера***

Таблицата показва поетапните резултати от изчисленията. В сиво е оцветен предишният етап. Разбира се, (a)=a и това не е отделено като „етап“.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| a | b | c | ca | ab(ca)bb | aa(ab(ca)bb)c |
| 0 | 1 | 1 | 1 | 01111=1111=011=11=0 | 0001=101=11=0 |
| 1 | 0 | 0 | 1 | 10100=1100=000=10=1 | 1110=010=10=1 |
| 0 | 1 | 0 | 1 | 01111=1111=011=11=0 | 0000=100=10=1 |
| 1 | 1 | 1 | 0 | 11011=0011=111=01=1 | 1111=011=11=0 |
| 0 | 0 | 1 | 1 | 00100=1100=000=10=1 | 0011=111=01=1 |