List comprehension

Time limit: 1.0s **Memory limit:** 512M

Minh là một học sinh lớp 11 đam mê lập trình và tự học ngôn ngữ Haskell trong thời gian rảnh. Một ngày nọ, khi đang tìm hiểu về list comprehension, cậu bỗng phát hiện ra một mẫu câu lệnh kỳ lạ trong một cuốn sách lập trình cổ điển mà cậu mượn được từ thư viện trường. Dòng mã này giống như một công thức thần bí, nó có thể sinh ra những dãy số kỳ lạ theo quy luật ẩn giấu.

Khi Minh nhập thử một đoạn code có dạng:

```
[(i+10, j) | i <- [3,4], j <- [5,6]]
```

Máy tính lập tức hiển thị:

```
[(13,5), (13,6), (14,5), (14,6)]
```

Cậu vô cùng phấn khích vì nhận ra rằng đây không chỉ là một công cụ mạnh mẽ để thao tác danh sách, mà còn có thể giúp cậu tái hiện lại cấu trúc toán học theo một cách hoàn toàn mới.

Hứng thú với phát hiện này, Minh chia sẻ điều đó với câu lạc bộ khoa học máy tính của trường. Để kiểm tra sự hiểu biết của Minh, thầy giáo đã đưa ra một thử thách:

`Em có thể viết một chương trình để phân tích một chuỗi đầu vào ở dạng list comprehension hai chiều đơn giản của Haskell và trả về danh sách kết quả tương ứng không?`

Thầy giáo yêu cầu chương trình của Minh có thể xử lý các chuỗi có dạng tổng quát:

```
[(f(i), f(j)) | i <- <list 1>, j <- <list 2>]
```

Trong đó:

- `list 1` và `list 2` có thể ở dạng $[m..n]$ (dãy số liên tiếp từ m đến n) hoặc $[a_1, a_2, \ldots, a_k]$ (danh sách số cụ thể).
- $f(i)$ và $f(j)$ có dạng `x` hoặc `<x opt y>` với:
  - `opt` là một trong các phép toán +, -, * (cộng, trừ, nhân).
  - x có thể là i hoặc j, còn y là một hằng số nguyên.

Xem thêm về tích Descartes để hình dung cơ chế của list comprehension dạng hai chiều.

Nếu Minh giải được bài toán này, cậu sẽ giành được một suất tham dự Hội thi Lập trình Viên Trẻ Toàn Quốc. Bạn hãy hỗ trợ cậu ấy cùng viết ra chương trình thú vị này nhé!

Input

Một dòng duy nhất là một biểu thức list comprehension hợp lệ của Haskell theo dạng:

```
[(f(i), f(j)) | i <- <list 1>, j <- <list 2>]
```

Trong đó:

- `list 1` và `list 2` có thể là:
  - $[m..n]$ (dãy số liên tiếp từ m đến n).
  - $[a_1, a_2, \ldots, a_k]$ (danh sách số cụ thể, phân tách bằng dấu phẩy).
- $f(i)$, $f(j)$ có dạng `x` hoặc `<x opt y>` với:
  - `opt` là một trong các phép toán +, -, *.
  - x có thể là i hoặc j, còn y là một hằng số nguyên.
- Dữ liệu đảm bảo tất cả các giá trị xuất hiện đều là các số nguyên không âm nhỏ hơn hoặc bằng 100.

Output

- Một dòng duy nhất chứa danh sách kết quả theo định dạng:

```
[(value1, value2), (value3, value4), ...]
```

- Mỗi cặp số (value*i*, *value*(i+1)) trong danh sách được sinh ra theo quy luật của list comprehension.

Scoring

- 50% số test thỏa mãn $f(i) = i$, $f(j) = j$. Đồng thời, `list 1` và `list 2` luôn có dạng $[a_1, a_2, \ldots, a_k]$.
- 50% số test còn lại không có ràng buộc gì thêm.

Ví dụ

Sample input 01

```
[(i+10, j) | i <- [3,4], j <- [5,6]]
```

Sample output 01

```
[(13,5), (13,6), (14,5), (14,6)]
```

Sample input 02

```
[(i, j) | i <- [1,2], j <- [2,2]]
```

Sample output 02

```
[(1,2), (1,2), (2,2), (2,2)]
```