BF 解释器

Brainfuck 是一种极小化的程序语言,可以模拟图灵机进行工作。它基于一个简单的机器模型,除了指令,这个机器还包括:一个以字节为单位、元素全部被初始化为零并且大小无限的数组、一个指向该数组的指针(初始时指向数组的第一个字节,后文称为"数据指针")以及用于输入输出的两个字节流。

基本知识

可将8个指令分为两类:数据操纵指令和控制流指令。

数据操纵指令定义如下:

指令	含义			
>	数据指针向右移动一个位置			
<	数据指针向左移动一个位置			
+	数据指针指向的字节的值加一			
-	数据指针指向的字节的值减一			
,	从输入流中读取一个字节,将其存入数据指针指向的位置			
	将数据指针指向的字节写入输出流中			

上述指令均为顺序执行。对于 , 指令 , 如果遇到 EOF , 应当将数据指针所指的字节的值改为 0.

控制流包含两个: [和], 这类指令不会影响数据指针和数据指针指向的字节的值, 只会影响下一条需要执行的指令。

指令	含义
	如果数据指针指向的单元值为 0 , 向后跳转到对应的] 指令的下一条指令处
1	如果数据指针指向的单元值不为 0 ,向前跳转到对应的 [指令的下一条指令处

输入描述

输入包含若干行,第一行为 BF 程序,其余行为该 BF 程序所需要的输入(若有)。

保证 BF 程序一定是有意义、正确且能够结束的。保证模拟的数组长度不大于 1000.

输出描述

输出所有 . 命令的结果。

示例

示例 1

读取一个字符并输出。

输入

```
, .
a
```

输出

a

示例 2

打印"Hello", 五个字符的 ASCII 码分别是 72 101 108 108 111.

输入

输出

Hello

注: 末尾没有换行符

解释

将上述 BF 程序第一个输出命令 . . 之前的命令取出分解为为如下

```
++++++++
[->+++++++<]
>.
```

在 +++++++ 之后数组变为如下状态:

ç	9	0	0	•••

[->+++++++-<] 的意思是:如果数据指针指向的字节的值不为 [0],将该值减 1,然后将数据指针右移一个位置,将当前位置加 8,数据指针再左移一个位置。其实就是循环了 9次,每次给数组第二个位置加 8,这段命令执行结束之后,数据指针指向第一个位置,数组如下:

0	72	0	•••

然后 > . 将数据指针右移一个位置(在此之前数据指针指向数组的第一个位置),输出当前的内容,即 72 , 是字母 H 的 ASCII 码。

提示

- 处理[和]命令不一定需要使用栈这种数据结构。
- 可使用如下方式读入 BF 程序(在 C++ 中, char 类型的变量可以用来表示一个字节,不过在输入以外的场景更推荐使用 <cstdint> 头文件中的 uint8_t):

```
#include <iostream>
#include <string>

// 读取 BF 程序
string cmd;
std::getline(std::cin, cmd);
```

可参考如下方式处理 , 指令:

```
#include <cstdint>

char c;
uint8_t byte;
if (!std::cin.get(c)) {
   byte = 0;
}
```