PPL Spring 2016 – Term Project

COMP 3220 Principle of Programming Languages

Due on Apr 30, 2016

Abstract

Implement an *interpreter function* for Brainfuck in `C++`.

1 Introduction

> Brainfuck, the ungodly creation of Urban Müller, whose goal was to create a Turing-complete language for which he could write the smallest compiler ever, for the Amiga OS 2.0 [1].

Brainfuck may be the most famous esoteric programming language, most of which are *useless* but *fun*. So is Brainfuck.

1.1 The Original Brainfuck

The Brainfuck programming language *ignores* all characters other than eight characters: `"+-.,<>[]"`, i.e., plus/minus sign, dot and comma, and angle/square bracket. Their effects are summarized in Table 1.

A Brainfuck program is basically a *collection of operations* on *an array* and *a pointer*. The array is **30000** bytes, all initialized to zero. And the pointer, which is free to move around within the array, is used to indicate which array cell to perform operation on. The pointer is initialized to point to the beginning of the array.

For example, one version of the "Hello, World!" program in Brainfuck

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

Table 1: The Brainfuck Programming Languages

Cmd	Effect	Equivalent in `C`
`+`	Increases element under pointer	`array[p]++`
`-`	Decreases element under pointer	`array[p]--`
`>`	Increases pointer	`p++`
`<`	Decreases pointer	`p--`
`[`	Starts loop, counter under pointer	`while(array[p]) {`
`]`	End of loop	`}`
`.`	Outputs ASCII code under pointer	`putchar(array[p])`
`,`	Reads char and stores under pointer	`array[p] = getchar()`

1.2 The Modified Brainfuck

There are some undefined behaviors in the original specification. What if

- an array cell overflows/underflows?
- the pointer goes out of bound?
- the character is non-printable?[1]
- the loop indicators, i.e., `[` and `]`, are unbalanced?
- and etc.

And more importantly, how can the TA test your implementation in batch? Based on the above considerations, I modified the Brainfuck as shown in Table 2. Note that only the modified commands are shown in Table 2, all other commands remain the same.

The modifications are explained in details.

Overflow/Underflow They are ignored. One (unsigned) byte can represent 0 through 255. If the byte is 0xFF, adding one with result in 0x00, i.e., the overflow bit is discarded. On the other hand, if the byte holds 0x00, subtracting 1 will result in 0xFF.

Out of Bound Pointer is wrapped around if out of bound. If it points to the end of the array, after being increased by 1, it will point to the beginning of the array. Similar for decreasing the pointer.

[1] Actually according to the author of Brainfuck, the dot command `.` should output the ASCII code under pointer. However his implementation used `putchar` which prints the character itself rather than its ASCII code.

Table 2: The Modified Brainfuck Programming Language

Cmd	Effect
`+`	Increases element under pointer, ignore overflow bit
`-`	Decreases element under pointer, ignore underflow bit
`>`	Increases pointer, wrapped around if out of bound
`<`	Decreases pointer, wrapped around if out of bound
`.`	Outputs ASCII code under pointer *to a byte stream*
`,`	Reads char *from a byte stream* and stores under pointer

Input/Output The interpreter reads from and writes to byte streams, instead of standard input/output stream (`stdin` and `stdout`). See the start-up code for more details.

Unbalanced Loop In case of unbalanced loop indicators, print out a warning message and your interpreter function should return false.

2 Requirement

Deliverable Zip the modified start-up code in a file, named `your_auburn_username.zip`, e.g., `zzg0009.zip`.

Submission This is an individual project. Please submit on Canvas before the deadline (**Apr 30, 2016**). *LATE SUBMISSION IS NOT ACCEPTED.*

3 Grading

You will find a couple of test cases in the start-up code. Your grade depends on the number of tests your code passes. The score S for term project is calculated as

$$S = 30 + 70 \times \frac{M}{N}$$

Where N is the total number of test cases, and M is the number of tests your code successfully passes. So long as you submit your code, you get at least 30 points. If you, however, intentionally submit an unfinished code or even empty project, you get ZERO.

References

[1] Brian Raiter. *Brainfuck. An Eight-Instruction Turing-Complete Programming Language*. June 2. URL: http://www.muppetlabs.com/~breadbox/bf/ (visited on 12/08/2015).