МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

РГЗ

по дисциплине: Математическая логика и теория алгоритмов
тема: «Интерпретатор языка Brainfuck»

Выполнил: ст. группы ПВ-211
Тутов Данил Андреевич

Проверили:
Куценко Дмитрий Александрович
Рязанов Юрий Дмитриевич

Белгород 2022 г.

Создание интерпретатора языка BrainFuck на языке C++

Задачи:

• Изучение синтаксиса языка BrainFuck.

• Реализация интерпретатора языка BrainFuck на языке C.

• Подбор тестовых данных для проверки работы программы.

Описание предметной области

Интерпретатор — программа, выполняющая интерпретацию.

Интерпретация — построчный анализ, обработка и выполнение исходного кода программы или запроса.

Brainfuck — один из известнейших эзотерических языков программирования, отличающийся особым минимализмом. Состоит из восьми команд, каждая из которых записывается одним символом.

Brainfuck был создан в 1993 году Урбаном Мюллером (Urban Müller), и авторская реализация остается фактическим стандартом языка. Его базовая спецификация достаточно проста, и он не был стандартизирован. Существует ряд любительских реализаций, ни одна из которых не представляет коммерческой ценности.

Brainfuck породил множество диалектов, отличающихся набором команд, способом их записи или деталями реализации, например, максимальным значением числа, хранящегося в ячейке, или ситуациями, вызывающими ошибку интерпретации. Brainfuck был изобретен при попытке создать Тьюринг-полный язык программирования с наименьшим возможным компилятором. Авторский компилятор имел размер 240 байт, а более поздние реализации достигают менее 200 байт.

Теоретически Brainfuck действительно обладает свойством Тьюринг-полноты и, следовательно, может выполнить любую задачу. На практике, впрочем, этот язык совершенно непригоден для решения задач из реальной жизни; выполнение даже простейших заданий становится вызовом для разработчика. Поэтому Brainfuck используется исключительно как математическая модель или для развлечения.

Brainfuck использует модель машины, напоминающую машину Тьюринга и состоящую из следующих элементов:

- программа — последовательность односимвольных команд языка и, возможно, других символов (при обработке игнорируются);

- указатель инструкций — указывает на команду, которая будет исполнена на следующем шагу, после ее исполнения передвигается на другую (обычно следующую справа) команду;

- память — моделируется одномерным массивом ячеек, в каждой ячейке хранится один байт; начальное значение ячейки — ноль;

- указатель данных — указывает на текущую ячейку памяти; начальное значение — самая левая ячейка массива; по команде двигается либо изменяет значение, хранящееся в текущей ячейке;

- потоки ввода и вывода — последовательности байтов в кодировке ASCII

Главная идея – манипулирование памятью. Вам выдается 30.000 массив 1 байтовых блоков, хотя на самом деле размер массива зависит от компилятора или интерпретатора, но стандартный размер - 30.000. Внутри этого массива вы можете увеличивать указатель, значение в ячейке и так далее. Для работы, как я уже говорил, используется 8 операторов:

> - увеличение указателя памяти или смещение право на 1 блок

< - уменьшение или смещение влево на 1 блок

+ - увеличение значения в ячейке памяти, на которую ссылается указатель

- - соответственно уменьшение на единиц

[- начало цикла, который выполняется пока текущая ячейка не ноль.

] - если значение в ячейке на которую указывает указатель не равно нулю, то переход на [

, - аналог getchar(), ввод одного символа

. - аналог putchar(), вывод одного символа на консоль

Программная реализация

```
#include <iostream>
#include <vector>

void interpretBrainFuckCode(std::vector<char> &acc, char *cpu) {
    unsigned int j = 0;
    int brc = 0;
    for (int i = 0; i < acc.size(); ++i) {
        switch (acc[i]) {
            case '>':
                j++;
                break;
            case '<':
                j--;
                break;
            case '+':
                cpu[j]++;
                break;
            case '-':
                cpu[j]--;
                break;
            case '.':
                std::cout << cpu[j];
                break;
            case ',':
                std::cin >> cpu[j];
                break;
            case '[':
                if (!cpu[j]) {
                    brc++;
                    while (brc) {
                        i++;
                        if (acc[i] == '[')
                            brc++;
                        if (acc[i] == ']')
                            brc--;
                    }
                } else {
                    continue;
                }
                break;
            case ']':
                if (!cpu[j]) {
                    continue;
                } else {
                    if (acc[i] == ']') {
                        brc++;
                    }
                    while (brc) {
                        i--;
                        if (acc[i] == '[') {
                            brc--;
                        }
                        if (acc[i] == ']') {
                            brc++;
                        }
                    }
                    i--;
                }
                break;
        }
    }
}
```

```
int main() {
    std::cout << "Input Brainfuck code: \n";

    char cpu[30000];
    std::vector<char> acc;
    char ch = getchar();

    while (ch != '\n') {
        acc.push_back(ch);
        ch = getchar();
    }

    interpretBrainFuckCode(acc, cpu);

    return 0;
}
```

Результат работы программы

```
Input Brainfuck code:
++++++++++[>+++++++>++++++++++>+++<<<-]>++.>+.+++++++..+++.>++.<<++++++++++++++.>.+++.------.--------.>+.
Hello World!
Process finished with exit code 0
```