Учебная виртуальная машина (УВМ)

Этот проект реализует ассемблер и интерпретатор для учебной виртуальной машины (УВМ), поддерживающей базовый набор команд. Ассемблер преобразует читаемый формат команд в бинарный файл, а интерпретатор выполняет команды из бинарного файла и создает дамп памяти в формате JSON.

Описание команд УВМ

Загрузка константы

Поле	Биты	Описание
A	Биты 0–3	Код команды (8)
B	Биты 4–30	Константа

Размер команды: 4 байта
Операнд: Поле B
Результат: Регистр-аккумулятор

Тест (A=8, B=600):

```
0x88, 0x25, 0x00, 0x00
```

Чтение значения из памяти

Поле	Биты	Описание
A	Биты 0–3	Код команды (10)
B	Биты 4–10	Смещение

Размер команды: 4 байта
Операнд: Значение в памяти по адресу, который является суммой адреса (регистр-аккумулятор) и смещения (поле B)
Результат: Регистр-аккумулятор

Тест (A=10, B=18):

```
0x2A, 0x01, 0x00, 0x00
```

Запись значения в память

Поле	Биты	Описание
A	Биты 0–3	Код команды (9)
B	Биты 4–14	Адрес

Размер команды: 4 байта
Операнд: Регистр-аккумулятор
Результат: Значение в памяти по адресу, который является полем B

Тест (A=9, B=85):

```
0x59, 0x05, 0x00, 0x00
```

Унарная операция: bswap()

Поле	Биты	Описание
A	Биты 0–3	Код команды (2)

Размер команды: 4 байта
Операнд: Значение в памяти по адресу, который является регистром-аккумулятором
Результат: Регистр-аккумулятор

Тест (A=2):

```
0x02, 0x00, 0x00, 0x00
```

Запуск проекта

1. **Ассемблер**
   Ассемблер принимает на вход файл с текстом исходной программы (задан из командной строки), генерирует бинарный файл с командами УВМ и сохраняет журнал команд в формате JSON.

2. **Интерпретатор**
   Интерпретатор принимает на вход бинарный файл и выполняет команды, сохраняя значения памяти в JSON формате.

Формат файлов

- **Файл-лог ассемблера**: JSON файл, содержащий ассемблированные инструкции в формате "ключ=значение".
- **Файл-результат интерпретатора**: JSON файл с дампом значений из диапазона памяти УВМ.

Тестирование

Для каждой команды предусмотрены тесты, которые проверяют корректность выполнения команд УВМ.

Исходная программа

```
set var0 600
set season1 1999
set season6 2006

mov premiere season1
mov finale season6

set to_bswap 43981                  ; 43981 = 0xABCD
set address_of_value_to_bswap 5     ; 5 because to_bswap is the 6th declared variable

bswap
```

Логи ассемблера

```
[{"A":8,"B":600,"hex":["0x88","0x25","0x0","0x0"]},
{"A":10,"B":0,"hex":["0x9","0x0","0x0","0x0"]},
{"A":8,"B":1999,"hex":["0xf8","0x7c","0x0","0x0"]},
{"A":10,"B":1,"hex":["0x19","0x0","0x0","0x0"]},
{"A":8,"B":2006,"hex":["0x68","0x7d","0x0","0x0"]},
{"A":10,"B":2,"hex":["0x29","0x0","0x0","0x0"]},
{"A":9,"B":43,"hex":["0xba","0x2","0x0","0x0"]},
{"A":10,"B":3,"hex":["0x39","0x0","0x0","0x0"]},
{"A":9,"B":51,"hex":["0x3a","0x3","0x0","0x0"]},
{"A":10,"B":4,"hex":["0x49","0x0","0x0","0x0"]},
{"A":8,"B":43981,"hex":["0xd8","0xbc","0xa","0x0"]},
{"A":10,"B":5,"hex":["0x59","0x0","0x0","0x0"]},
{"A":8,"B":5,"hex":["0x58","0x0","0x0","0x0"]},
{"A":10,"B":6,"hex":["0x69","0x0","0x0","0x0"]},
{"A":2,"hex":["0x2","0x0","0x0","0x0"]}]
```


Результат (Дамп памяти)


```
{
  "AC": 52651,
  "MEMORY": [
    {
      "0b00000000000": 600
    },
    {
      "0b00000000001": 1999
    },
    {
      "0b00000000010": 2006
    },
    {
      "0b00000000011": 1999
    },
    {
      "0b00000000100": 2006
    },
    {
      "0b00000000101": 43981
    },
    {
      "0b00000000110": 5
    },
```