virtual-machine-sisop

Trabalho da cadeira de Sistemas Operacionais - PUCRS 2020/2

TODOS na classe APP do programa

Integrantes

- Marcelo de Souza
- Camila Borba
- Gustavo Vidaletti
- Georgia Antunes

Programas

Todos os programas solicitações foram feitos, não houve nenhum alteração nas instruções *(seguimos o modelo inicial proposto)*. Os programas se encontram no diretório `src/main/resources/files/`

- `program0` - Sequência fibonacci dos 10 primeiros números
- `program1` - Sequência fibonacci de um valor **x** lido de uma posição da memória
- `program2` - fatoriade um valor **x** lido de uma posição da memória
- `program3` - Bubble sort ordena 10 posições.

Nenhum apresentou problemas a principio, foram criados testes para cada um deles e seus casos.

VM

Instruções para rodar a VM

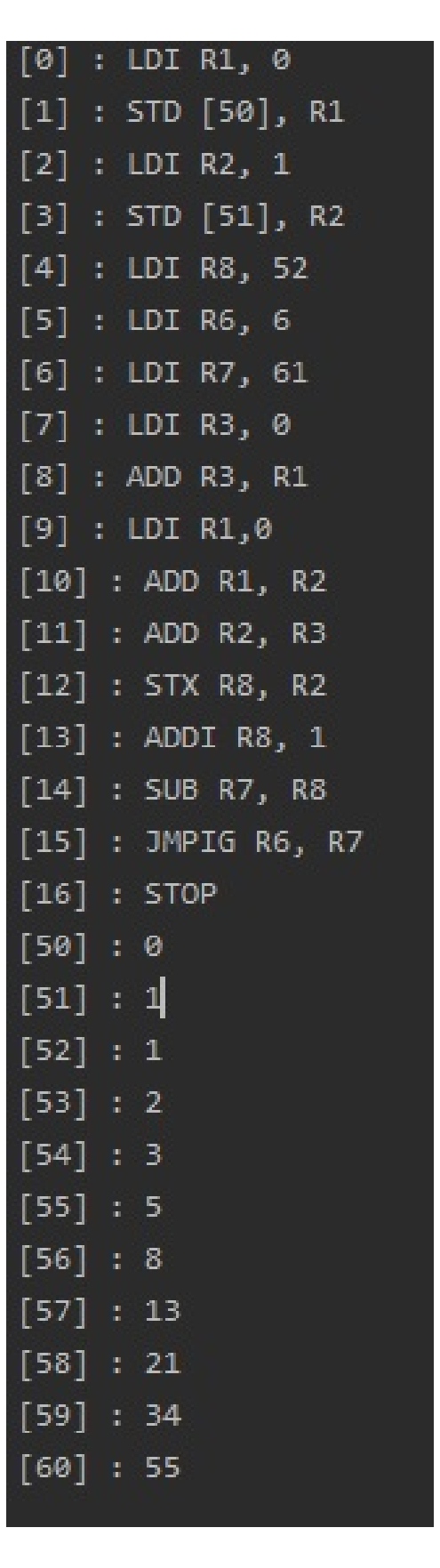
- Versão do java necessário: `11 no minimo`
- Clonar o projeto e abrir em alguma IDE de preferência
- Abrir a pasta root do projeto como projeto `maven`
- Rodar o seguinte comando: `mvn clean install`

Testes dos programas

- No diretório `src/test/java/com/trabalho/sisop/` encontrasse a classe `VirtualMachineTest`
- Executar cada teste separadamente, pois devido a problema de concorrência dos mesmos, acabam quebrando.
- A cada execução de teste, é possível verificar o log de toda a execução do programa e as posicões de memória que foram populadas pela execução do programa.

resultados testes

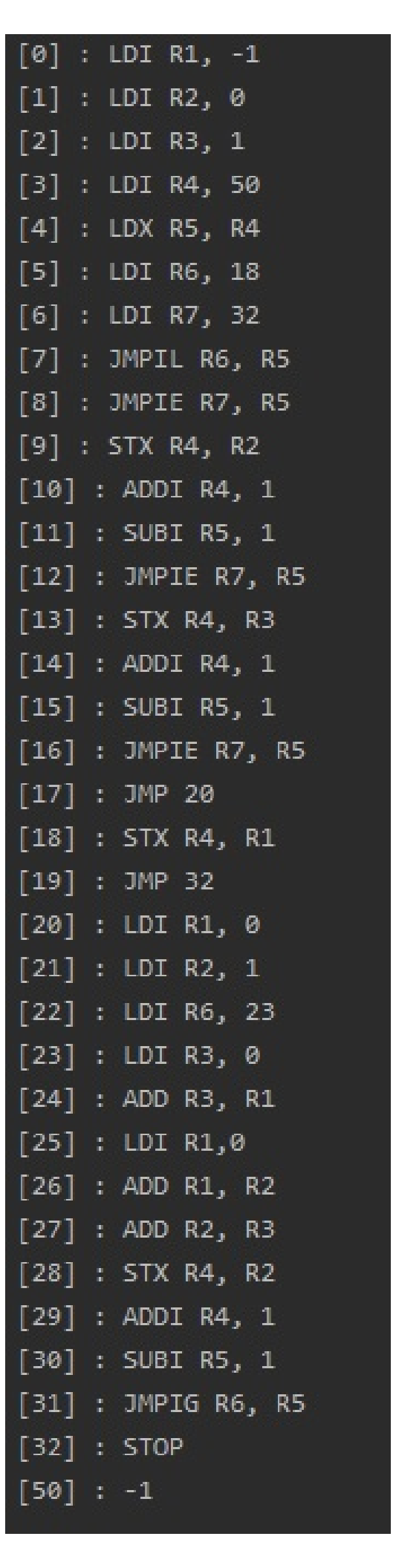
teste_programa_1



```
[0] : LDI R1, 0
[1] : STD [50], R1
[2] : LDI R2, 1
[3] : STD [51], R2
[4] : LDI R8, 52
[5] : LDI R6, 6
[6] : LDI R7, 61
[7] : LDI R3, 0
[8] : ADD R3, R1
[9] : LDI R1,0
[10] : ADD R1, R2
[11] : ADD R2, R3
[12] : STX R8, R2
[13] : ADDI R8, 1
[14] : SUB R7, R8
[15] : JMPIG R6, R7
[16] : STOP
[50] : 0
[51] : 1
[52] : 1
[53] : 2
[54] : 3
[55] : 5
[56] : 8
[57] : 13
[58] : 21
[59] : 34
[60] : 55
```

teste_programa_2_caso_valor_negativo

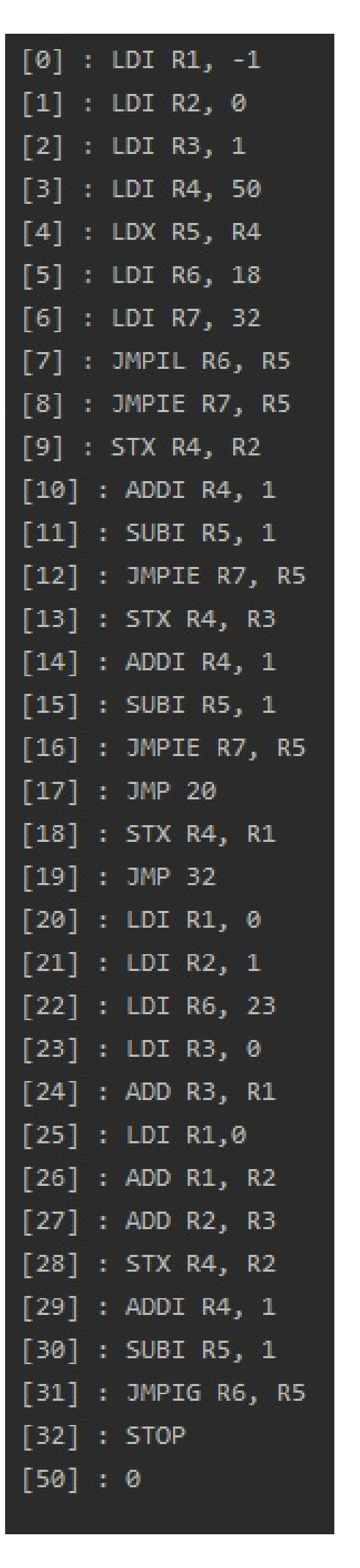
- Sequência fibonacci de um valor negativo



```
[0] : LDI R1, -1
[1] : LDI R2, 0
[2] : LDI R3, 1
[3] : LDI R4, 50
[4] : LDX R5, R4
[5] : LDI R6, 18
[6] : LDI R7, 32
[7] : JMPIL R6, R5
[8] : JMPIE R7, R5
[9] : STX R4, R2
[10] : ADDI R4, 1
[11] : SUBI R5, 1
[12] : JMPIE R7, R5
[13] : STX R4, R3
[14] : ADDI R4, 1
[15] : SUBI R5, 1
[16] : JMPIE R7, R5
[17] : JMP 20
[18] : STX R4, R1
[19] : JMP 32
[20] : LDI R1, 0
[21] : LDI R2, 1
[22] : LDI R6, 23
[23] : LDI R3, 0
[24] : ADD R3, R1
[25] : LDI R1,0
[26] : ADD R1, R2
[27] : ADD R2, R3
[28] : STX R4, R2
[29] : ADDI R4, 1
[30] : SUBI R5, 1
[31] : JMPIG R6, R5
[32] : STOP
[50] : -1
```

teste_programa_2_caso_valor_igual_zero

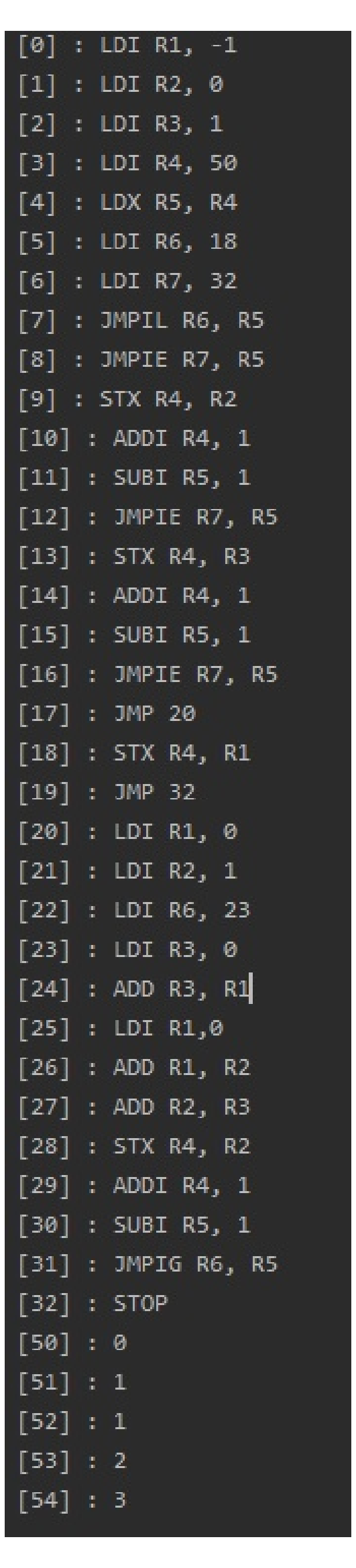
- Sequência fibonacci de 0



```
[0] : LDI R1, -1
[1] : LDI R2, 0
[2] : LDI R3, 1
[3] : LDI R4, 50
[4] : LDX R5, R4
[5] : LDI R6, 18
[6] : LDI R7, 32
[7] : JMPIL R6, R5
[8] : JMPIE R7, R5
[9] : STX R4, R2
[10] : ADDI R4, 1
[11] : SUBI R5, 1
[12] : JMPIE R7, R5
[13] : STX R4, R3
[14] : ADDI R4, 1
[15] : SUBI R5, 1
[16] : JMPIE R7, R5
[17] : JMP 20
[18] : STX R4, R1
[19] : JMP 32
[20] : LDI R1, 0
[21] : LDI R2, 1
[22] : LDI R6, 23
[23] : LDI R3, 0
[24] : ADD R3, R1
[25] : LDI R1,0
[26] : ADD R1, R2
[27] : ADD R2, R3
[28] : STX R4, R2
[29] : ADDI R4, 1
[30] : SUBI R5, 1
[31] : JMPIG R6, R5
[32] : STOP
[50] : 0
```

teste_programa_2_caso_valor_maior_que_zero

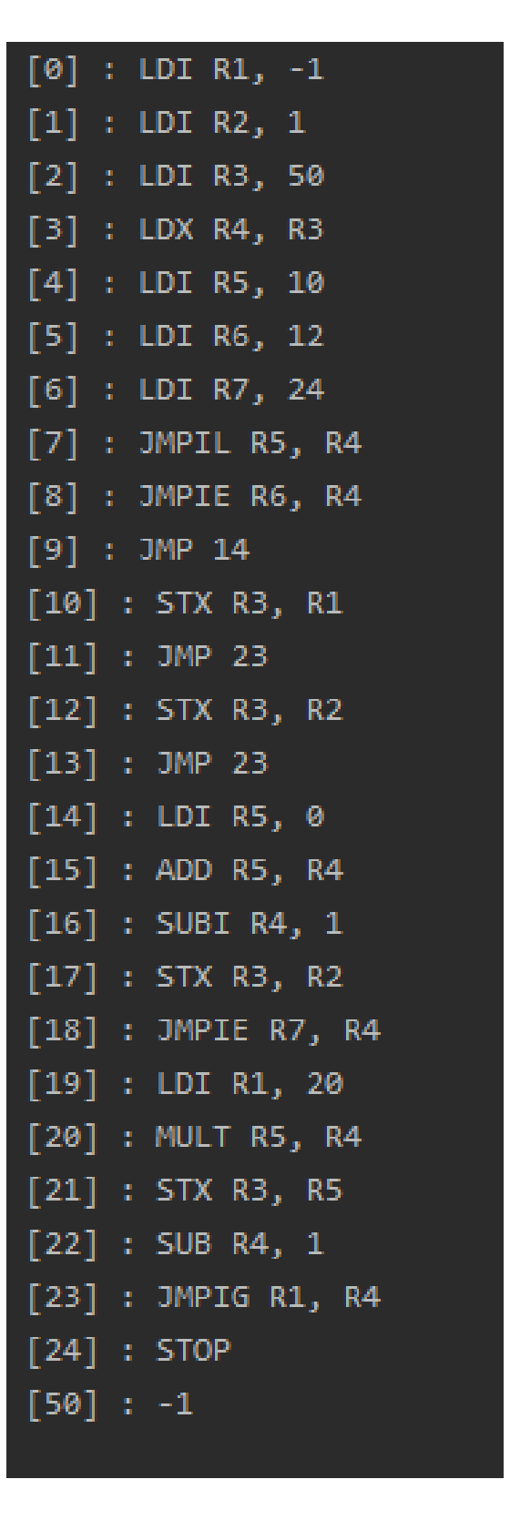
- Sequência fibonacci de 5 números



```
[0] : LDI R1, -1
[1] : LDI R2, 0
[2] : LDI R3, 1
[3] : LDI R4, 50
[4] : LDX R5, R4
[5] : LDI R6, 18
[6] : LDI R7, 32
[7] : JMPIL R6, R5
[8] : JMPIE R7, R5
[9] : STX R4, R2
[10] : ADDI R4, 1
[11] : SUBI R5, 1
[12] : JMPIE R7, R5
[13] : STX R4, R3
[14] : ADDI R4, 1
[15] : SUBI R5, 1
[16] : JMPIE R7, R5
[17] : JMP 20
[18] : STX R4, R1
[19] : JMP 32
[20] : LDI R1, 0
[21] : LDI R2, 1
[22] : LDI R6, 23
[23] : LDI R3, 0
[24] : ADD R3, R1
[25] : LDI R1,0
[26] : ADD R1, R2
[27] : ADD R2, R3
[28] : STX R4, R2
[29] : ADDI R4, 1
[30] : SUBI R5, 1
[31] : JMPIG R6, R5
[32] : STOP
[50] : 0
[51] : 1
[52] : 1
[53] : 2
[54] : 3
```

teste_programa_3_caso_valor_negativo

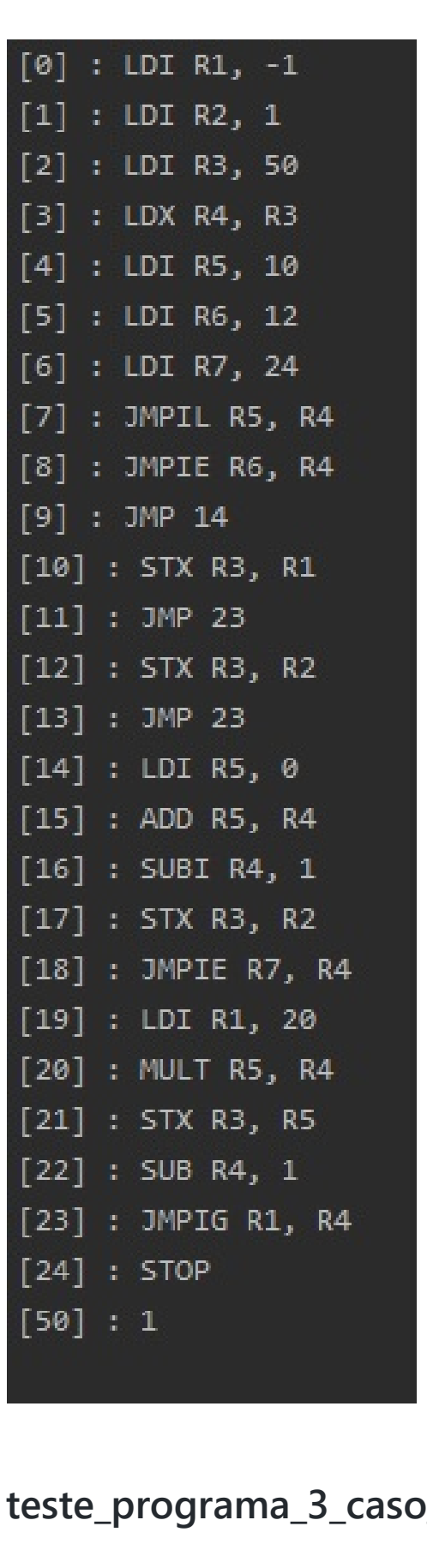
- Fatorial de um número negativo



```
[0] : LDI R1, -1
[1] : LDI R2, 1
[2] : LDI R3, 50
[3] : LDX R4, R3
[4] : LDI R5, 10
[5] : LDI R6, 12
[6] : LDI R7, 24
[7] : JMPIL R5, R4
[8] : JMPIE R6, R4
[9] : JMP 14
[10] : STX R3, R1
[11] : JMP 23
[12] : STX R3, R2
[13] : JMP 23
[14] : LDI R5, 0
[15] : ADD R5, R4
[16] : SUBI R4, 1
[17] : STX R3, R2
[18] : JMPIE R7, R4
[19] : LDI R1, 20
[20] : MULT R5, R4
[21] : STX R3, R5
[22] : SUB R4, 1
[23] : JMPIG R1, R4
[24] : STOP
[50] : -1
```

teste_programa_3_caso_valor_igual_zero

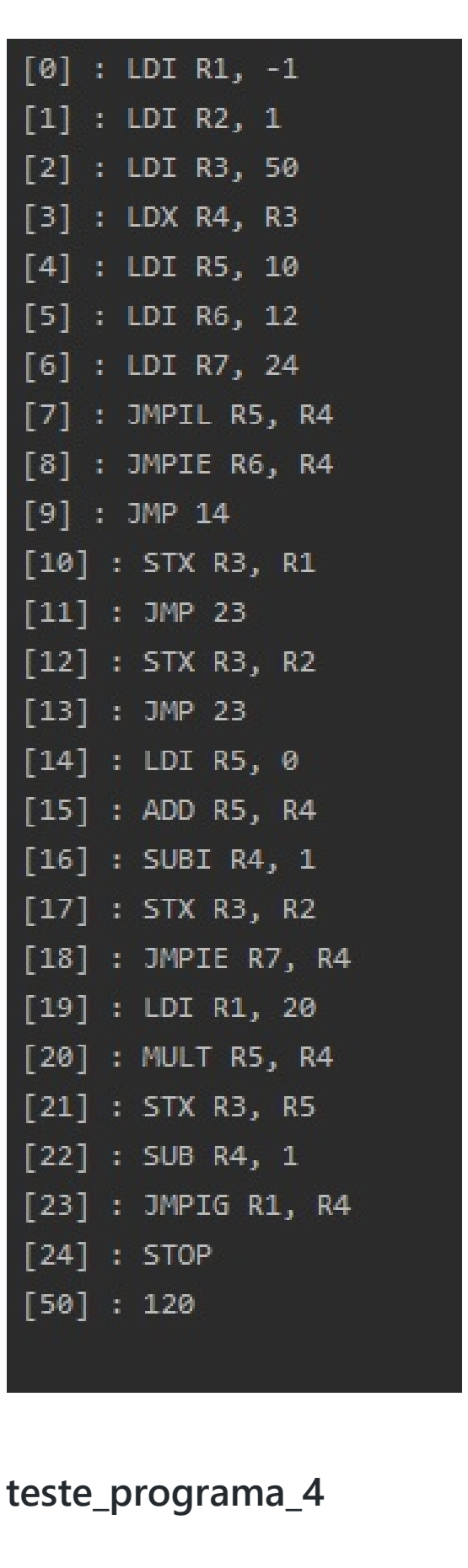
- Fatorial de 0



```
[0] : LDI R1, -1
[1] : LDI R2, 1
[2] : LDI R3, 50
[3] : LDX R4, R3
[4] : LDI R5, 10
[5] : LDI R6, 12
[6] : LDI R7, 24
[7] : JMPIL R5, R4
[8] : JMPIE R6, R4
[9] : JMP 14
[10] : STX R3, R1
[11] : JMP 23
[12] : STX R3, R2
[13] : JMP 23
[14] : LDI R5, 0
[15] : ADD R5, R4
[16] : SUBI R4, 1
[17] : STX R3, R2
[18] : JMPIE R7, R4
[19] : LDI R1, 20
[20] : MULT R5, R4
[21] : STX R3, R5
[22] : SUB R4, 1
[23] : JMPIG R1, R4
[24] : STOP
[50] : 1
```

teste_programa_3_caso_valor_maior_que_zero

- Fatorial do valor 5

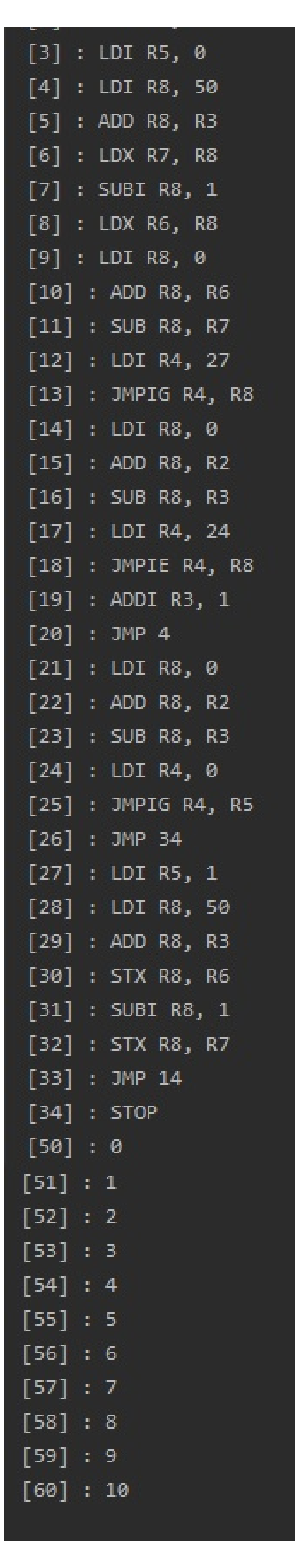


```
[0] : LDI R1, -1
[1] : LDI R2, 1
[2] : LDI R3, 50
[3] : LDX R4, R3
[4] : LDI R5, 10
[5] : LDI R6, 12
[6] : LDI R7, 24
[7] : JMPIL R5, R4
[8] : JMPIE R6, R4
[9] : JMP 14
[10] : STX R3, R1
[11] : JMP 23
[12] : STX R3, R2
[13] : JMP 23
[14] : LDI R5, 0
[15] : ADD R5, R4
[16] : SUBI R4, 1
[17] : STX R3, R2
[18] : JMPIE R7, R4
[19] : LDI R1, 20
[20] : MULT R5, R4
[21] : STX R3, R5
[22] : SUB R4, 1
[23] : JMPIG R1, R4
[24] : STOP
[50] : 120
```

teste_programa_4

- A carga de dados nesse teste foi feita da seguinte maneira

```
memory.writeValueToMemory(50, 10);
memory.writeValueToMemory(51, 9);
memory.writeValueToMemory(52, 8);
memory.writeValueToMemory(53, 7);
memory.writeValueToMemory(54, 6);
memory.writeValueToMemory(55, 5);
memory.writeValueToMemory(56, 4);
memory.writeValueToMemory(57, 3);
memory.writeValueToMemory(58, 2);
memory.writeValueToMemory(59, 1);
memory.writeValueToMemory(60, 0);
```



```
 [3] : LDI R5, 0
 [4] : LDI R8, 50
 [5] : ADD R8, R3
 [6] : LDX R7, R8
 [7] : SUBI R8, 1
 [8] : LDX R6, R8
 [9] : LDI R8, 0
 [10] : ADD R8, R6
 [11] : SUB R8, R7
 [12] : LDI R4, 27
 [13] : JMPIG R4, R8
 [14] : LDI R8, 0
 [15] : ADD R8, R2
 [16] : SUB R8, R3
 [17] : LDI R4, 24
 [18] : JMPIE R4, R8
 [19] : ADDI R3, 1
 [20] : JMP 4
 [21] : LDI R8, 0
 [22] : ADD R8, R2
 [23] : SUB R8, R3
 [24] : LDI R4, 0
 [25] : JMPIG R4, R5
 [26] : JMP 34
 [27] : LDI R5, 1
 [28] : LDI R8, 50
 [29] : ADD R8, R3
 [30] : STX R8, R6
 [31] : SUBI R8, 1
 [32] : STX R8, R7
 [33] : JMP 14
 [34] : STOP
 [50] : 0
[51] : 1
[52] : 2
[53] : 3
[54] : 4
[55] : 5
[56] : 6
[57] : 7
[58] : 8
[59] : 9
[60] : 10
```

Observações

- A VM conta com uma memória que é representada por um array de Strings.
- Carrega as instruções do programa no inicio da memória.
- Carrega os dados apartir da posição **[50]** de memória.
- Os programas `program1`, `program2`, `program3` não fazem nenhuma carga de valores na memória para serem executados, como o `program0`, portando se for executado o programa sem ser pelos testes criados **(que popula a memória para executar cada caso de cada programa)**, será necessário popular os valores na mão.
- Qualquer dúvida ou dificuldade em executar o programa, e também para outras finalidades como tirar dúvidas sobre o código fonte da VM, favor entrar em contato com o grupo.