



APRESENTANDO O
SIMULADOR MIC-1

- **Yago do Nascimento Santos**
- **Pedro Henrique Neves Sixel**
- **João Pedro Saliveros Cordeiro**
- **Rafael Ribeiro Ramos Pinto**

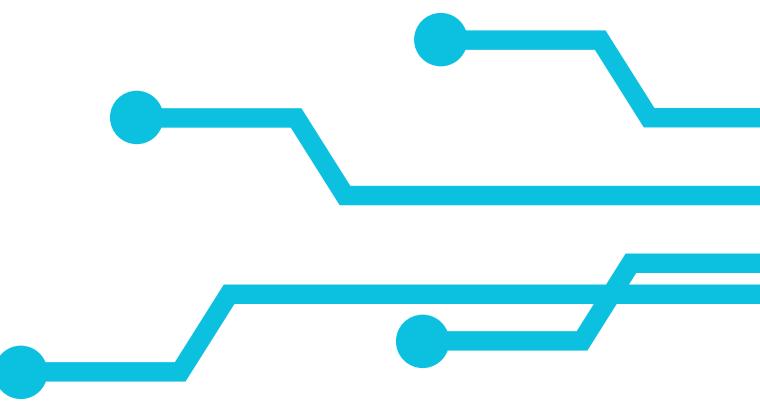




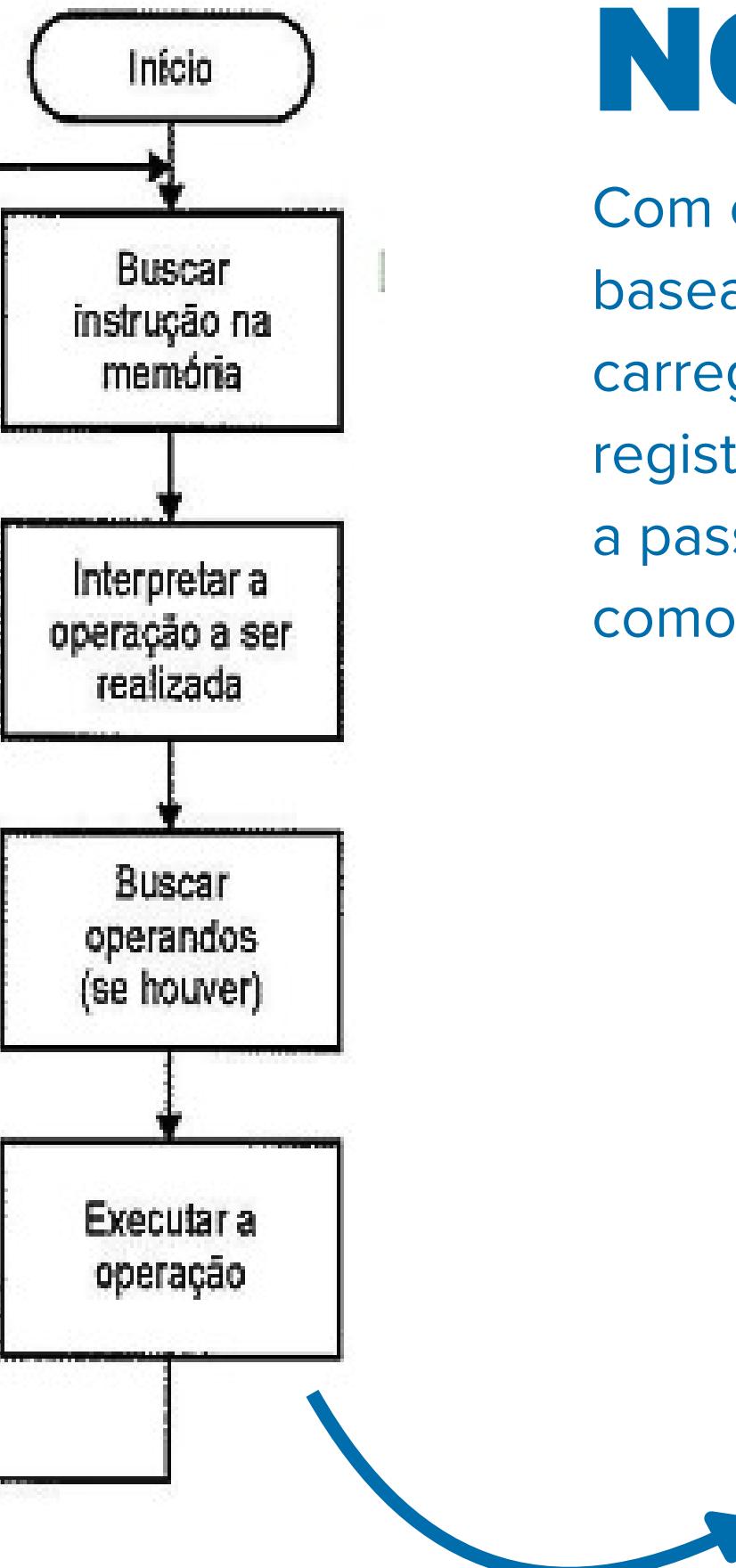
SUMÁRIO

- Objetivo
- Linguagem de Implementação
- Classe do Simulador, Registradores e Memória
- Instruções
- Exemplo de Execução
- Fim da Apresentação

NOSSO OBJETIVO



Com o intuito de representar didaticamente a operação de um processador simples, baseado no modelo MIC-1, nosso simulador oferece uma interface que permite carregar e executar programas, convertidos em instruções. Ele dispõe de registradores, memória e instruções, e o usuário consegue visualizar a execução passo a passo. Isso torna mais fácil entender o fluxo de controle, as operações aritméticas e como os dados são manipulados na memória.



The screenshot displays the MIC-1 simulator interface with the following details:

Carregar Arquivo: C:/Users/Joaó Pedro/Downloads/teste1.txt

Estado da CPU:

- PC: 0x000C
- AC: 0x0000 (0)
- SP: 0x0FFF
- MAR: 0x0032
- MBR: 0x0001
- IR: 0x09
- Next Inst.: HALT
- Flags: Z=True N=False

Code (teste1.txt):

```
1 INICIO: LOCO 1
2      STOD 50    / Mem[50] = 1
3
4
5
6
7 LOOP: JZER 12
8      SUBD 50
9      JUMP 6
10
11 FIM: HALT
12
13
```

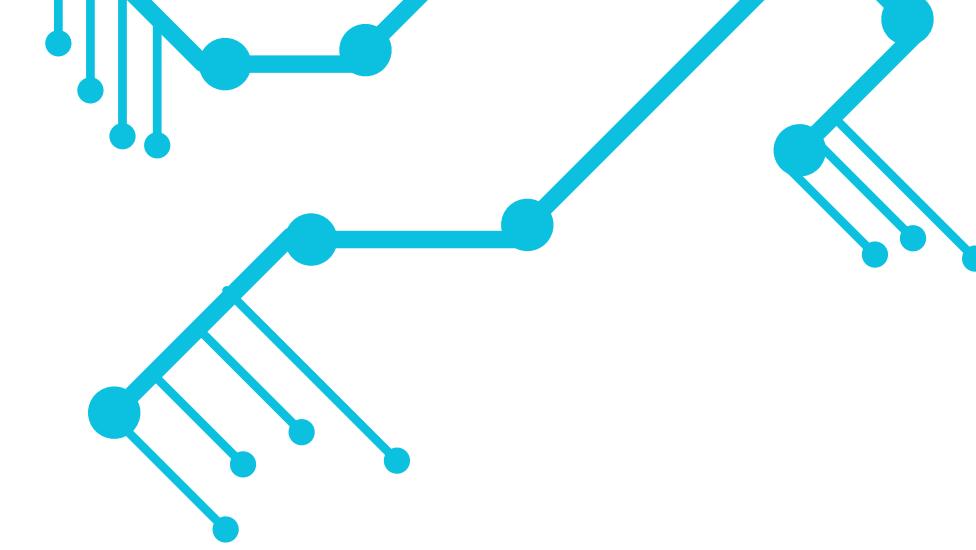
Status: Passo executado. PC=12

Buttons: Reset, Passo, Executar, Pausar, Delay (ms): 500



LINGUAGEM DE IMPLEMENTAÇÃO

Nosso programa é desenvolvido em Python, uma escolha estratégica devido à sua simplicidade técnica e à disponibilidade de recursos amplamente conhecidos para aprendizado e pesquisa. Ferramentas, como o Tkinter que foi empregado na interface gráfica, exemplificam essa acessibilidade.



- **DEMONSTRAÇÃO DO INÍCIO DA CLASSE SIMULADOR**
- **DEFININDO O TAMANHO DA MEMÓRIA**
- **DEFININDO OS REGISTRADORES**

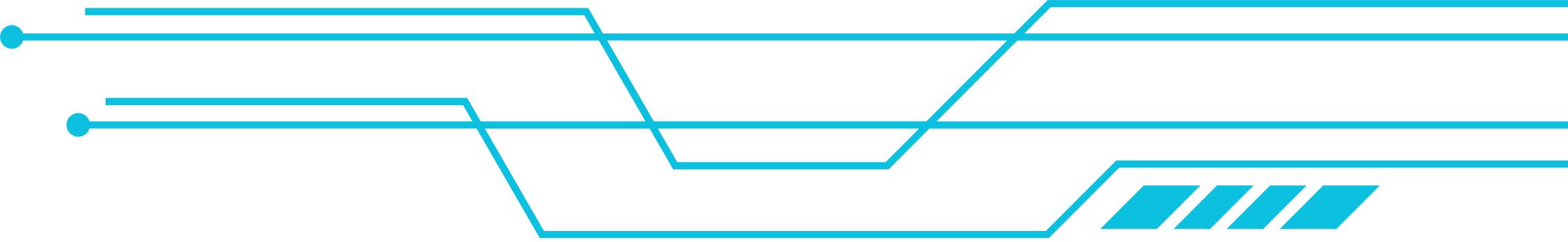
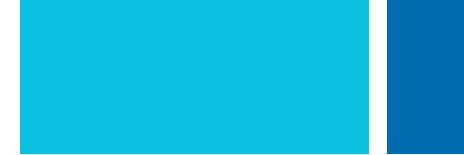
```
#Simulador MIC-1
class MIC1Simulator: 2 usages
    def __init__(self):
        self.tamanhoDaMemoria = 4096
        self.memoria = [0] * self.tamanhoDaMemoria

        #Registradores
        self.pc = 0 #Program Counter
        self.ac = 0 #Accumulador
        self.sp = self.tamanhoDaMemoria - 1 #Stack Pointer
        self.mar = 0 #Registrador de End. de Mem.
        self.mbr = 0 #Memory Buffer
        self.ir = 0 #Registrador de instrução
```

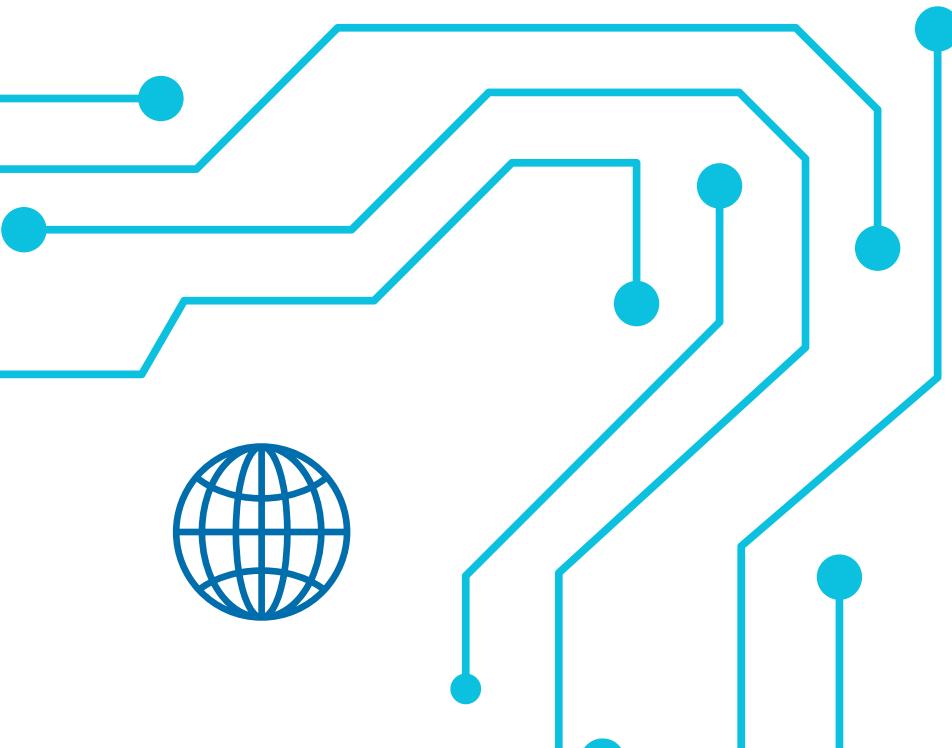


DEFININDO OS OPCODES DE CADA INSTRUÇÃO

```
#Instruções
self.instructions = {
    0x00: ("HALT", self.halt),
    0x01: ("LOAD", self.load),
    0x02: ("STOD", self.store),
    0x03: ("ADDD", self.add),
    0x04: ("SUBD", self.sub),
    0x05: ("JPOS", self.jpos),
    0x06: ("STODL", self.stodl),
    0x07: ("LOCO", self.loadcons),
    0x08: ("JUMP", self.jump),
    0x09: ("JZERO", self.jumpZERO),
    0x0A: ("STODL", self.jneg),
    0x0B: ("ADDL", self.call),
    0x0C: ("INSP", self.insp),
    0x0D: ("PUSH", self.push),
    0x0E: ("POP", self.pop),
}
```



EXEMPLO DE EXECUÇÃO



**Vamos utilizar o
seguinte arquivo
de texto como
instruções:**

```
teste.txt

    / Inicializacao: Cria a constante 1 na memoria (endereco 50)
INICIO: LOCO 1
        STOD 50      / Mem[50] = 1

    / Configura o contador inicial no AC (comeca com 4)
    LOCO 4

    / --- INICIO DO LOOP (Endereco de memoria 6) ---
LOOP:   JZER 12      / Se AC for Zero, pula para o final (endereco
        12)

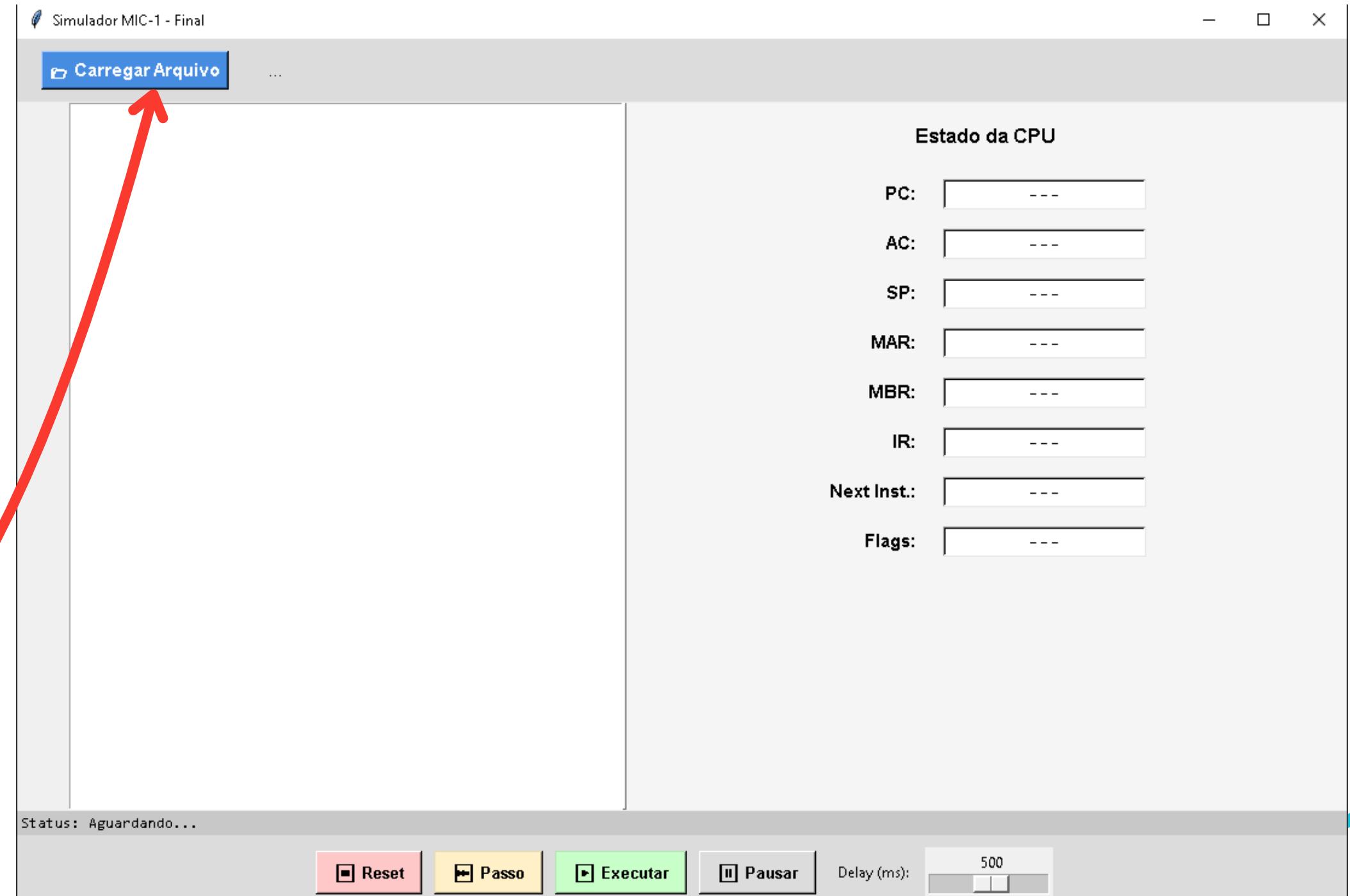
        SUBD 50      / Subtrai o valor que esta no endereco 50 (4 -
        1)

        JUMP 6       / Volta para o inicio do LOOP (endereco 6)

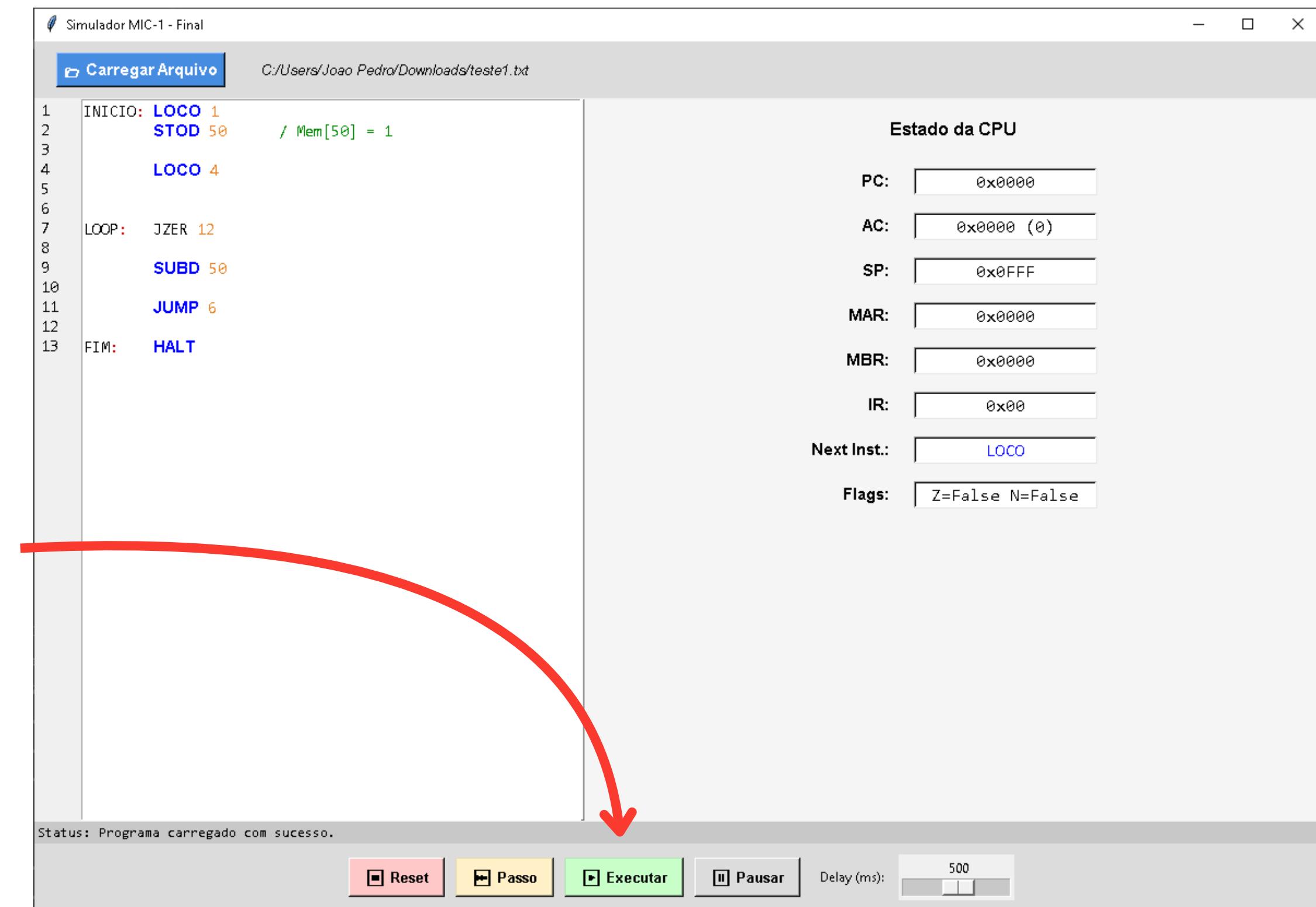
    / --- FIM (Endereco de memoria 12) ---
FIM:    HALT
```



**Ao abrir o simulador,
precisamos carregar
um arquivo .txt com
as instruções escritas**



**Com o arquivo
carregado, podemos
iniciar a execução.**



**Inicialização: Cria a
constante 1 na
memoria (endereço
50)**

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joa Pedro/Downloads/teste1.txt

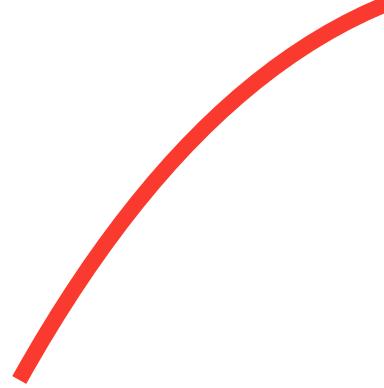
```
1 INICIO: LOCO 1
2           / Mem[50] = 1
3
4
5
6
7 LOOP: JZER 12
8
9 SUBD 50
10
11 JUMP 6
12
13 FIM: HALT
```

Estado da CPU

PC:	0x0000
AC:	0x0000 (0)
SP:	0x0FFF
MAR:	0x0000
MBR:	0x0000
IR:	0x00
Next Inst.:	LOCO
Flags:	Z=False N=False

Status: Programa carregado com sucesso.

Delay (ms):



**Configura o contador
inicial no AC (começa
com 4)**

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joao Pedro/Downloads/teste1.txt

```
1 INICIO: LOCO 1
2           STOD 50 / Mem[50] = 1
3
4 LOCO 4
5
6
7 LOOP: JZER 12
8
9 SUBD 50
10
11 JUMP 6
12
13 FIM: HALT
```

Estado da CPU

PC:	0x0006
AC:	0x0004 (4)
SP:	0x0FFF
MAR:	0x0032
MBR:	0x0001
IR:	0x07
Next Inst.:	JZER
Flags:	Z=False N=False

Status: Passo executado. PC=6

Reset Passo Executar Pausar Delay (ms): 500

INICIO DO LOOP
**(Endereço de
memória 6)**
**Se AC for Zero, pula
para o final
(endereço 12)**

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joa Pedro/Downloads/teste1.txt

1 INICIO: LOCO 1	2 STOD 50 / Mem[50] = 1	Estado da CPU
3	4	PC: 0x0008
5	6 LOCO 4	AC: 0x0004 (4)
7	8 LOOP: JZER 12	SP: 0x0FFF
9	10 SUBD 50	MAR: 0x0032
11	12 JUMP 6	MBR: 0x0001
13 FIM: HALT	IR: 0x09	
	Next Inst.: SUBD	Flags: Z=False N=False

Status: Passo executado. PC=8

Delay (ms): 500



**Subtrai o valor que
esta no endereço 50
(4 - 1)**

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joao Pedro/Downloads/teste1.txt

```
1 INICIO: LOCO 1
2 STOD 50 / Mem[50] = 1
3
4 LOCO 4
5
6
7 LOOP: JZER 12
8 SUBD 50
9
10 JUMP 6
11
12 FIM: HALT
```

Estado da CPU

PC:	0x0008
AC:	0x0003 (3)
SP:	0x0FFF
MAR:	0x0032
MBR:	0x0001
IR:	0x09
Next Inst.:	SUBD
Flags:	Z=False N=False

Status: Passo executado. PC=8

Reset Passo Executar Pausar Delay (ms): 500

**Volta para o inicio do
LOOP (endereço 6)**

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joao Pedro/Downloads/teste1.txt

```
1 INICIO: LOCO 1
2           STOD 50 / Mem[50] = 1
3
4           LOCO 4
5
6
7 LOOP:   JZER 12
8
9           SUBD 50
10          JUMP 6
11
12          FIM: HALT
```

Estado da CPU

PC:	0x0008
AC:	0x0003 (3)
SP:	0x0FFF
MAR:	0x0032
MBR:	0x0001
IR:	0x09
Next Inst.:	SUBD
Flags:	Z=False N=False

Status: Passo executado. PC=8

Delay (ms):

**Repete o LOOP até
AC = 0**

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joao Pedro/Downloads/teste1.txt

```
1 INICIO: LOCO 1
2 STOD 50 / Mem[50] = 1
3
4
5
6 LOOP: JZER 12
7
8 SUBD 50
9
10 JUMP 6
11
12 FIM: HALT
```

Estado da CPU

PC:	0x0008
AC:	0x0003 (3)
SP:	0x0FFF
MAR:	0x0032
MBR:	0x0001
IR:	0x09
Next Inst.:	SUBD
Flags:	Z=False N=False

Status: Passo executado. PC=8

Delay (ms): 500

AC = 0, pula para o final (endereço 12)

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joaoo Pedro/Downloads/teste1.txt

1 INICIO: LOCO 1	2 STOD 50 / Mem[50] = 1	Estado da CPU
3	4	PC: 0x000C
5	6 LOCO 4	AC: 0x0000 (0)
7	8 LOOP: JZER 12	SP: 0x0FFF
9	10 SUBD 50	MAR: 0x0032
11	12 JUMP 6	MBR: 0x0001
13 FIM: HALT	IR: 0x09	
	Next Inst.: HALT	Flags: Z=True N=False

Status: Passo executado. PC=12

Delay (ms): 500

Fim do programa

Simulador MIC-1 - Final

Carregar Arquivo C:/Users/Joao Pedro/Downloads/teste1.txt

1 INICIO: LOCO 1	2 STOD 50 / Mem[50] = 1	Estado da CPU
3	4	PC: 0x000C
5	6	AC: 0x0000 (0)
7 LOOP: JZER 12	8	SP: 0xFFFF
9	10 SUBD 50	MAR: 0x0032
11	12 JUMP 6	MBR: 0x0001
13 FIM: HALT	14	IR: 0x09
	15	Next Inst.: HALT
	16	Flags: Z=True N=False

Status: Passo executado. PC=12

Reset Passo Executar Pausar Delay (ms): 500

FIM DA APRESENTAÇÃO

Obrigado pela atenção de todos!