Pontifícia Universidade Católica do Rio Grande do Sul

FELIPE FREITAS SILVA, LÁZARO MACIEL VASCONCELOS E LUIZA HELLER KROEFF PLÁ

RELATÓRIO: TRABALHO 2

Porto Alegre, Rio Grande do Sul

2022

**Sumário**

[1.Questão 1 3](#_Toc118382957)

[1.1 Enunciado da Questão 1 3](#_Toc118382958)

[1.2 Solução 4](#_Toc118382959)

[1.3 Explicação Passo a passo do Código Acima: 8](#_Toc118382960)

[1.4 Testes de Verificação 10](#_Toc118382961)

[1.4.1Teste 1: frase: “Primeiro” 10](#_Toc118382962)

[1.4.2Teste 2: frase: “Teste de 4 palavras” 12](#_Toc118382963)

[1.4.3Teste 3: frase: “Frase de teste muito grande mesmo com mais de 10 palavras” 14](#_Toc118382964)

[2.Questão 2 17](#_Toc118382965)

[2.1 Enunciado da Questão 2 17](#_Toc118382966)

[2.2 Solução em Código Assembly Viking 17](#_Toc118382967)

[2.3 Explicação Passo a passo do Código Acima: 21](#_Toc118382968)

[2.4 Testes de Verificação 23](#_Toc118382969)

[2.4.1Teste 1: segmento 0 – 2 23](#_Toc118382970)

[2.4.2Teste 2: 4 – 8 25](#_Toc118382971)

[2.4.3Teste 3: 0 – 11 27](#_Toc118382972)

[REFERÊNCIAS 29](#_Toc118382973)

# 1.Questão 1

## 1.1 Enunciado da Questão 1

Escreva um programa que conta o número de palavras armazenadas em uma String e apresenta o total no terminal. Como sugestão, utilize a função abaixo como referência para sua implementação, a ser chamada a partir do programa principal. Utilize duas Strings na demonstração do funcionamento.

int count\_words(char \*str, int size) {

int i = 0, words = 0;

while (1) {

while ((str[i] < 33) || (str[i] > 126)) {

if (i >= size)

return words;

i++;

}

while ((str[i] > 32) && (str[i] < 127)) {

if (i >= size)

return words + 1;

i++;

}

words++;

}

}

## 1.2 Solução

main

; r2 = totalPalavras(começa em 0)

ldr r2,0

; r5 -> &frase

ldi r5,frase

; salva r5 na pilha

sub sp,2

stw r5,sp

; r6 -> &count

ldi r6,count

rep

; pega r5 da pilha

ldw r5,sp

add sp,2

; r4 = r5[i] (primeiro byte da frase)

ldb r4,r5

; se não tiverem mais caracteres por ler, goTo(final)

bez r4,print\_final

; printa o byte em r4

stw r4,0xf000

; r5 -> &mensagem + 1 (próximo byte)

add r5,1

; salva r5 na pilha

sub sp,2

stw r5,sp

; r3 = 33 (primeiro caracter depois do " " na tabela ASCII)

ldr r3,33

; r3 = r4 < r3 ? 1 : 0

slt r3,r4,r3

; se r4 for um espaço, goTo(count)

bnz r3,r6

; r3 = 126 (último caracter considerado como letra no exercício)

ldr r3,126

; r3 = r4 < r3 ? 1 : 0

slt r3,r4,r3

; se r4 não for uma letra, goTo(count)

bez r3,r6

; repete o laço

bnz r7,rep

count

; totalPalavras++

add r2,1

; volta para o laço de repetição

bnz r7,rep

print\_final

; r4 -> &mensagem

ldi r4,mensagem

print\_rep

; r5 = r4[0] (primeiro byte da mensagem)

ldb r5,r4

; print(r5)

stw r5,0xf000

; r4 -> &mensagem + 1 (próximo byte)

add r4,1

; se r5 não for mais caractere da frase, goTo(final)

bez r5,final

; enquanto r5 for um caractere da frase, repete o laço

bnz r5,print\_rep

final

; totalPalavras++

add r2,1

; print(totalPalavras)

stw r2,0xf002

hcf

frase "Teste com 4 palavras"

mensagem "\nTotal de palavras:\t"

## 1.3 Explicação Passo a passo do Código Acima:

Código do método de rótulo: main:

1. Carrega-se o registrador r2 com o valor 0.
2. Carrega-se o endereço da variável mensagem no registrador r5.
3. Se vai para a próxima posição livre no registrador de armazenamento r7.
4. Carrega-se o valor do registrador r5, passo 2, na posição livre acessada.
5. O registrador r6 aponta para o método count.

Código do método de rótulo: rep:

1. Carrega-se o valor de r5 na posição livre do registrador de armazenamento r7.
2. Acessa-se a próxima posição livre do registrador de armazenamento.
3. Carrega-se o primeiro byte do registrador r5 no registrador r4.
4. Se o valor de r4 for igual a 0, ou seja, não for uma letra, ocorre um jump para o método de rótulo: print\_final.
5. Printa-se o byte armazenado no registrador r4.
6. Adiciona-se 1 ao registrador r5, ou seja, aponta para a próxima letra ou espaço.
7. Acessa-se a próxima posição livre do registrador de armazenamento.
8. Armazena-se o valor de r5 em r7.
9. Guarda-se no registrador r3 o caractere de valor 33 da tabela ASCII.
10. Se o byte guardado em r4 for menor do que o byte guardado em r3, então o registrador r3 recebe o valor 1.
11. Se r3 for diferente de 0, em outras palavras, se o passo 14 for verdade, ocorre um jump para o endereço armazenado em r6 (método: count).
12. Guarda-se no registrador r3 o caractere de valor 126 da tabela ASCII.
13. Se o byte guardado em r4 for menor do que o byte guardado em r3, então o registrador r3 recebe o valor 1.
14. Se r3 for igual a 0, ou seja, o passo 17 for falso, ocorre um jump para o endereço armazenado em r6 (método: count).
15. Ocorre repetição do laço.

Código do método de rótulo: count:

1. Adiciona-se 1 ao registrador r2, ou seja, achou-se mais uma palavra.
2. Repetição do laço (de rótulo: rep).

Código do método de rótulo: print\_final

1. O registrador r4 aponta para o endereço da variável mensagem.

Código do método de rótulo: print\_rep

1. R5 vai receber o primeiro byte da mensagem, que está no registrador r4.
2. Printa o valor do registrador r5 (o próximo caractere da mensagem).
3. R4 aponta para o próximo byte da mensagem.
4. Se o registrador r5 não encontrar mais nenhum byte da mensagem, ocorre um jump para o método de rótulo: final.
5. Se o registrador r5 encontrar um byte de caractere na mensagem, ocorre repetição do laço.

Código do método de rótulo: final:

1. Conta mais uma palavra.
2. Printa o total de palavras.
3. Halt and catch fire.

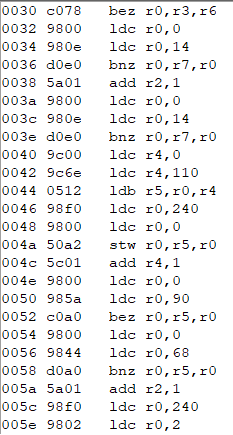
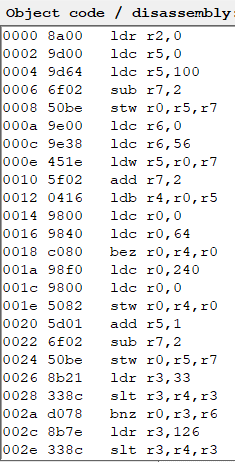
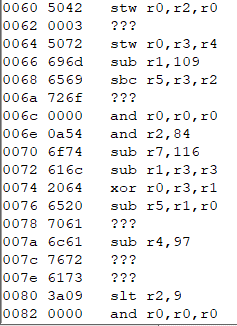
## 1.4 Testes de Verificação

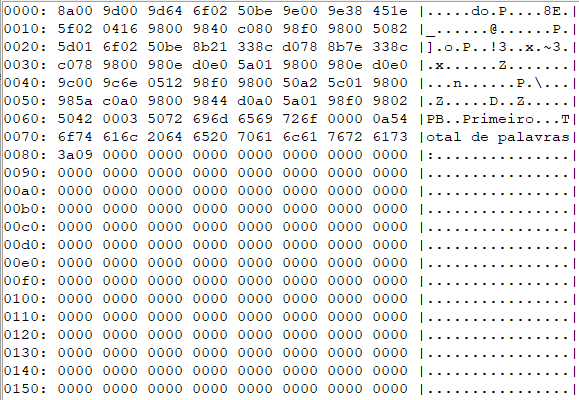
### 1.4.1Teste 1: frase: “Primeiro”

Texto

Descrição gerada automaticamenteTabela de símbolos:

Código de máquina:



Dump de memória:

Saída no terminal:

Texto

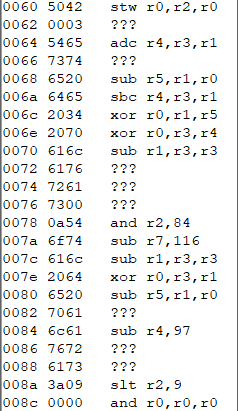
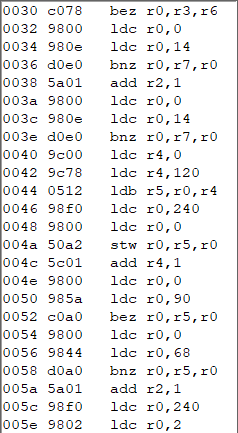
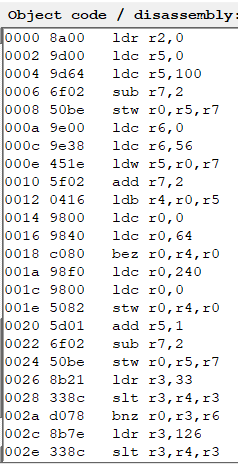
Descrição gerada automaticamente com confiança baixa

### 1.4.2Teste 2: frase: “Teste de 4 palavras”

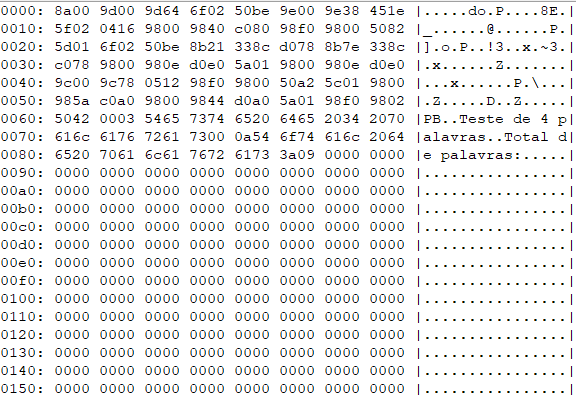
Texto

Descrição gerada automaticamente Tabela de símbolos:

Código de máquina:



Dump da memória:



Saída no terminal:

Uma imagem contendo Texto

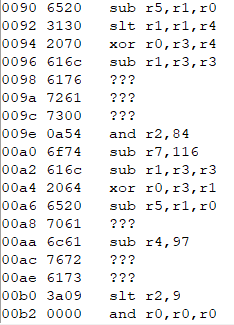
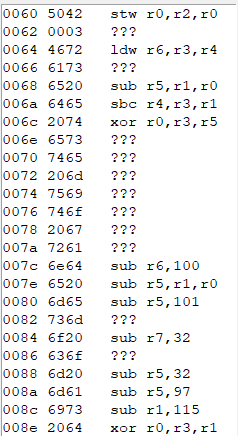
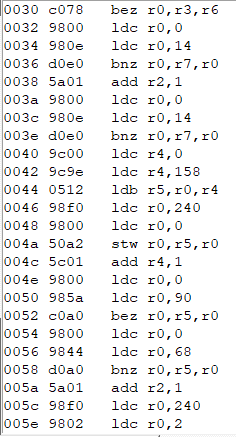
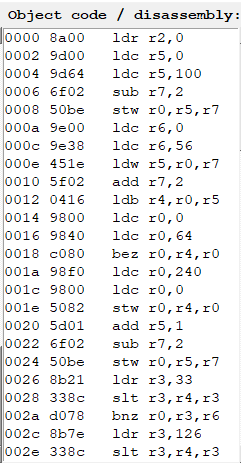
Descrição gerada automaticamente

### 1.4.3Teste 3: frase: “Frase de teste muito grande mesmo com mais de 10 palavras”

Texto

Descrição gerada automaticamenteTabela de símbolos:

Código de máquina:



Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente com confiança médiaDump da memória:

Saída no terminal:

Texto

Descrição gerada automaticamente

# 2.Questão 2

## 2.1 Enunciado da Questão 2

Considere uma sequência de n números inteiros. Para esta sequência, determine um segmento de soma máxima e o valor dessa soma. Por exemplo, para a sequência de valores 5, 2, -2, -7, 3, 14, 10, -3, 9, -6, 4, 1, o segmento está entre o índices 4 e 8 e a soma dos valores desse segmento ´e 33. Utilize dois vetores de números na demonstração do funcionamento da sua solução.

## 2.2 Solução em Código Assembly Viking

main

; r1 = soma(começa em 0)

; r2 = sequencia

ldi r2,sequencia

; r3 à segmento

ldi r3,segmento

; r4 = segmento[0]

ldw r4,r3

; r5 = segmento[1]

add r3,2

ldw r5,r3

; salva r1,r2

sub sp,2

stw r1,sp

sub sp,2

stw r2,sp

; r1 = r2 = 0

ldi r1,0

ldi r2,0

; r1 = r5 - r4

sub r1,r5,r4

; r6 = r1 (quantas vezes repito o laco)

xor r6,r1,r2

; recupera r1,r2

ldw r2,sp

add sp,2

ldw r1,sp

add sp,2

; r2 -> sequence[i-1]

sub r2,2

; r5 = -1

ldi r5,-1

move\_pointer

; r2 -> sequence[i+1]

add r2,2

; r3 = 0

ldi r3,0

; r4--;

sub r4,1

; if (r4 >= 0) {

slt r3,r5,r4

; repeat }

bnz r3,move\_pointer

; r2 -> sequence[i-1]

sub r2,2

laco\_soma

; r2 -> sequence[i+1]

add r2,2

; r3 = 0

ldi r3,0

; r3 = sequencia[i]

ldw r3,r2

; r1 += r3

add r1,r3,r1

; r6--

sub r6,1

; if (r6 >= 0) {

slt r4,r5,r6

; repeat }

bnz r4,laco\_soma

atr\_msg

; r5 -> mensagem[0]

ldi r5,mensagem

print\_str

; r4 = mensagem[i]

ldb r4,r5

; print(r4)

stw r4,0xf000

; r5 -> mensagem[i+1] (goToNextChar)

add r5,1

; repeat while there are words

bnz r4,print\_str

print\_number

stw r1,0xf002

hcf

sequencia 5 2 -2 -7 3 14 10 -3 9 -6 4 1

segmento 4 8

mensagem "Soma: "

## 2.3 Explicação Passo a passo do Código Acima:

1. R1 recebe uma soma que começa em 0, enquanto r2 recebe uma sequência numérica, indicada na variável ao final do código.
2. Carrega-se a sequência para o registrador r2.
3. Carrega-se o segmento (entre os índices 4 e 8) no registrador r3.
4. O registrador r4 recebe o valor do segmento de índice 0 (de valor 4).
5. O registrador r3 passa a apontar para o próximo índice do segmento.
6. O registrador r5 recebe o valor do segmento de índice 1 (de valor 8).
7. A próxima posição livre na pilha é acessada.
8. Guarda-se o valor de r1 na posição livre do registrador de armazenamento (r7).
9. Acessa-se a próxima posição livre na pilha.
10. Guarda-se o valor do registrador r2 na posição livre acessada.
11. Carrega-se o valor zero no registrador r1.
12. Carrega-se o valor zero no registrador r2.
13. Realiza-se uma subtração dos valores armazenados em r5 e r4, armazenando o resultado no registrador r1 (8-4=4).
14. O valor de r1 é passado para r6, e esse valor indica a quantidade de vezes que o laço será realizado.
15. Carrega-se o valor da última posição do registrador de armazenamento (índice 1 = 8) no registrador r2.
16. Acessa-se a próxima posição do registrador de armazenamento.
17. Carrega-se o valor da posição acessada na última instrução no registrador r1.
18. Acessa-se o próximo valor livre do registrador r7. (Instruções realizadas para recuperar os registradores r1 e r2.)
19. O registrador r2 aponta para o índice i-1 da sequência.
20. Carrega-se o valor -1 no registrador r5.

**Código do método de rótulo: move\_pointer:**

1. O registrador r2 aponta para o índice i+1 da sequência. (ou seja, volta para o 0).
2. Carrega-se o valor 0 no registrador r3.
3. O valor r4 irá sofrer a subtração: r4 = r4 – 1 (O valor da subtração será mantido em r4).
4. Se o valor do registrador r5 for menor que o valor do registrador r4, o registrador r3 recebe 1.
5. Se o registrador r3 não receber 0, ou seja, se r5 for realmente menor que r4, ocorre um jump para o método move\_pointer, logo repetindo o laço.
6. O registrador r2 aponta para o índice i-1 da sequencia.

**Código do método de rótulo: laço\_soma:**

1. O registrador r2 aponta para o índice i+1 da sequência.
2. Carrega-se o valor 0 no registrador r3.
3. R3 recebe o valor da posição de índice que r2 está apontando.
4. R1 recebe o valor de r3 + r1.
5. R6 irá sofrer uma subtração: r6 -1, e esse resultado será armazenado no registrador r6.
6. Se r5 for menor que r6, r4 recebe 1, caso contrário r4 recebe 0. (em outras palavras, se r6 for maior ou igual a 0).
7. Se r4 não for 0 ocorre um jump para o método laço\_soma e repete-se o laço.

**Código do método de rótulo: atr\_msg:**

1. R5 aponta para o endereço de mensagem (S).

**Código do método de rótulo: print\_str:**

1. Carrega-se o valor do registrador r5 no registrador r4, em outras palavras r4 recebe o caractere.
2. Printa-se o valor armazenado em r4.
3. R5 aponta para o índice i+1 da variável mensagem (o).
4. Enquanto palavras existirem ocorre um jump para o método print\_str, repetindo o laço, ou seja, imprime o próximo caractere.

**Código do método de rótulo: print\_number:**

1. Printa-se o valor armazenado no registrador r1 (o resultado da soma).
2. Halt and catch fire.

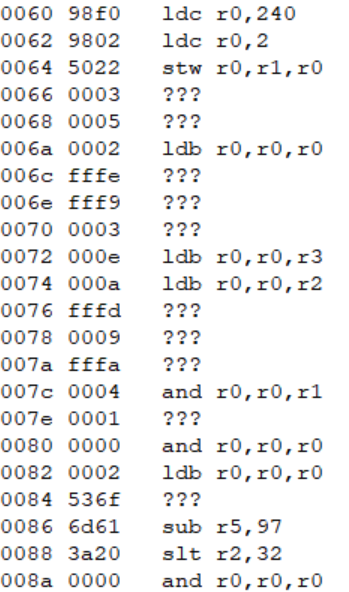
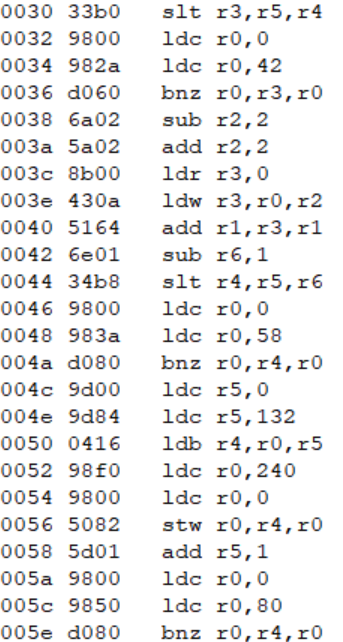
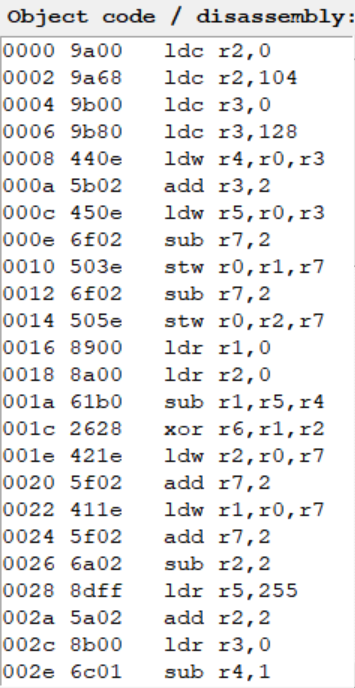
## 2.4 Testes de Verificação

### 2.4.1Teste 1: segmento 0 – 2

Texto

Descrição gerada automaticamenteTabela de Símbolos:

Código de Máquina:



Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente com confiança médiaDump da Memória:

Texto

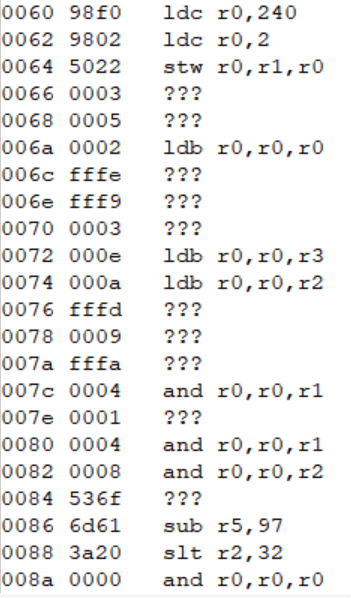
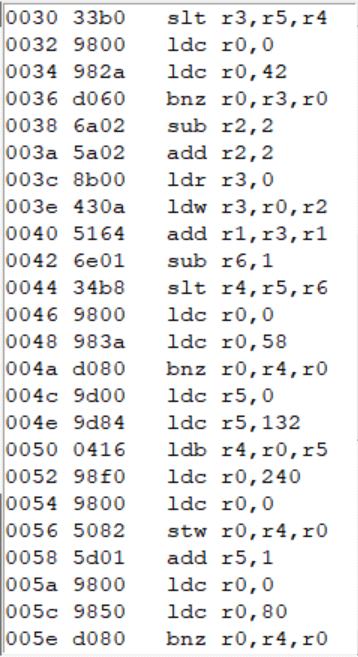
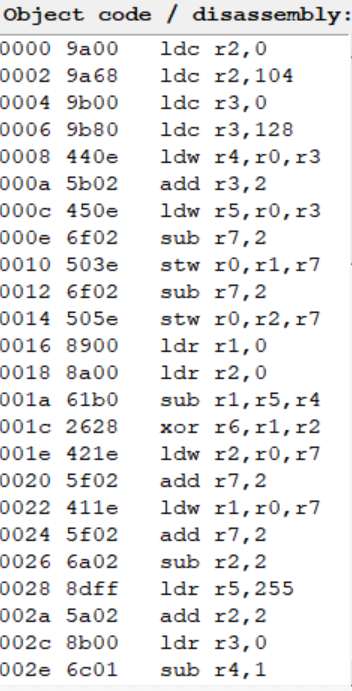
Descrição gerada automaticamente com confiança médiaSaída no Terminal:

### 2.4.2Teste 2: 4 – 8

Texto, Carta

Descrição gerada automaticamenteTabela de Símbolos:

Código de Máquina:



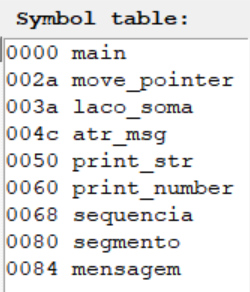
Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente com confiança médiaDump da Memória:

Texto

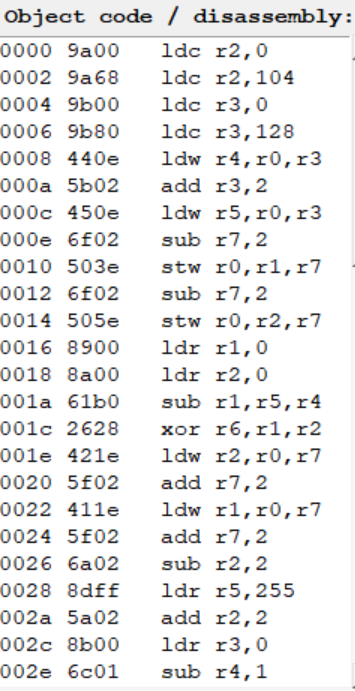
Descrição gerada automaticamente com confiança médiaSaída no Terminal:

### 2.4.3Teste 3: 0 – 11

Tabela de Símbolos:

Tabela

Descrição gerada automaticamenteTabela

Descrição gerada automaticamenteCódigo de Máquina:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente com confiança médiaDump da Memória:

Texto

Descrição gerada automaticamenteSaída no Terminal:

# REFERÊNCIAS

FILHO, Sérgio Johann. Viking CPU - **Manual de referência v0.5**, dez. 2018.