Universidade Estadual de Londrina

Arquitetura de Computadores II – 1COP0012

Atividades Práticas de Laboratório n. 1

Data de entrega: 06/03/2022

O programa exemplo abaixo faz a leitura de 5 valores inteiros e os armazena em um vetor de inteiros. Após o armazenamento é realizada a impressão dos valores. O programa está estruturado utilizando **um procedimento de leitura** e **um procedimento de escrita**.

```
1  .data
2  ent: .asciiz "Insira o valor de Vet["
3  ent2: .asciiz "]: "
4  .align 2
5  vet: .space 20
6
7  .text
8
9  main: la $a0, vet # Endereço do vetor como parâmetro
10       jal leitura # leitura(vet)
11       move $a0, $v0 # Endereço do vetor retornado
12       jal escrita # escrita(vet)
13       li $v0, 10 # Código para finalizar o programa
14       syscall # Finaliza o programa
15
16 leitura:
17    move $t0, $a0 # Salva o endereço base de vet
18    move $t1, $t0 # Endereço de vet[i]
19    li $t2, 0 # i = 0
20 l: la $a0, ent # Carrega o endereço da string
21    li $v0, 4 # Código de impressão de string
22    syscall # Impressão da string
23    move $a0, $t2 # Carrega o índice do vetor
24    li $v0, 1 # Código de impressão de inteiro
25    syscall # Imprime o índice i
26    la $a0, ent2 # Carrega o endereço da string
27    li $v0, 4 # Código de impressão de string
28    syscall # Impressão da string
29    li $v0, 5 # Código de leitura de inteiro
30    syscall # Leitura do valor
31    sw $v0, ($t1) # Salva o valor lido em vet[i]
32    add $t1, $t1, 4 # Endereço de vet[i+1]
33    addi $t2, $t2, 1 # i++
34    blt $t2, 5, l # if(i < 5) goto l
35    move $v0, $t0 # Endereço de vet para retorno
36    jr $ra # Retorna para a main
```

```
37
38  escrita:
39     move $t0, $a0 # Salva o endereço base de vet
40     move $t1, $t0 # Endereço de vet[i]
41     li $t2, 0 # i = 0
42  e: lw $a0, ($t1) # Carrega o valor de vet[i]
43     li $v0, 1 # Código de impressão de inteiro
44     syscall # Imprime vet[i]
45     li $a0, 32 # Código ASCII para espaço
46     li $v0, 11 # Código de impressão de caractere
47     syscall # Imprime um espaço
48     add $t1, $t1, 4 # Endereço de vet[i+1]
49     addi $t2, $t2, 1 # i++
50     blt $t2, 5, e # if(i < 5) goto e
51     move $v0, $t0 # Endereço de vet para retorno
52     jr $ra # Retorna para a main
```

Exercícios

1) Elaborar um programa, em código MIPS, que faça a leitura de um vetor de n elementos inteiros e execute, **utilizando procedimentos**, as seguintes operações:

a) Ordene o vetor em ordem crescente e apresentar o vetor ordenado;

b) Realize a soma dos elementos pares do vetor e apresentar o valor;

c) Leia uma chave k (número inteiro) e apresente na saída o número de elementos do vetor que são maiores que a chave k e menores que 2*k;

d) Leia uma chave k (número inteiro) e apresente na saída o número de elementos iguais a chave lida.

e) Apresenta na saída o resultado da soma dos números inteiros perfeitos menos a soma dos números inteiros semiprimos.

2) Elaborar um programa, em código MIPS, que faça a leitura de um vetor de n elementos inteiros e uma chave k. Na saída apresente o vetor com os elementos deslocados de k posições a direita (rotacionar).

Obs. pesquisar a definição de número perfeito e semiprimo.

Universidade Estadual de Londrina

Arquitetura de Computadores II – 1COP0012

Atividades Práticas de Laboratório n. 2

Data de entrega: 06/03/2022

Exemplo de tradução de código C utilizando *malloc()* para código MIPS.

```
int *n = malloc(sizeof(int));
*n = 3;
int *vet = malloc(sizeof(int) * 10);
vet[0] = 7;
vet[3] = 11;
vet[8] = 34;
char *s = malloc(sizeof(char) * 20);
scanf("%s", s);
```

```
li $a0, 4 # 4 bytes (inteiro)
li $v0, 9 # Código de alocação dinâmica heap
syscall # Aloca 4 bytes (endereço em $v0)
move $t0, $v0 # Move para $t0
li $t1, 3 # aux = 3
sw $t1, ($t0) # *n = 3
li $a0, 40 # 40 bytes (espaço para 10 inteiros)
li $v0, 9 # Código de alocação dinâmica heap
syscall # Aloca 40 bytes
move $t1, $v0 # Move para $t1
li $t2, 7 # aux = 7
sw $t2, ($t1) # v[0] = 7
li $t2, 11 # aux = 11
sw $t2, 12($t1) # v[3] = 11
li $t2, 34 # aux = 34
sw $t2, 32($t1) # v[8] = 34
li $a0, 20 # 20 bytes (espaço para 20 char)
li $v0, 9 # Código de alocação dinâmica heap
syscall # Aloca 20 bytes
move $a0, $v0 # Endereço base da string
li $a1, 20 # Número máximo de caracteres
li $v0, 8 # Código para leitura de string
syscall # scanf("%s", s)
```

Utilizando alocação dinâmica em MIPS, implementar **os seguintes códigos**:

1) Elaborar um programa que faça a leitura de dois vetores (**VetA** e **VetB**) compostos, cada um, de **n elementos inteiros** e apresente como saída a somatória dos elementos das posições pares de **VetA** subtraída da somatória dos elementos das posições ímpares de **VetB**.

2) Desenvolver um programa que faça a leitura de um vetor **Vet**, de **n elementos inteiros**, e apresente como saída o maior e o menor elementos do vetor e suas respectivas posições (primeira posição = 1).

3) Faça um programa que leia dois vetores (**VetC** e **VetD**), de **n elementos inteiros**, e apresente como saída outro vetor (**VetE**) contendo nas posições pares os valores do primeiro e nas posições impares os valores do segundo.

4) Elaborar um programa que leia um vetor **Vet**, de **n elementos inteiros**, e o "compacte", ou seja, elimine as posições com valor igual a zero. Para isso, todos os elementos à frente do valor zero devem ser movidos uma posição para trás do vetor. Apresente como saída o vetor compactado (**Vetcomp**).

Universidade Estadual de Londrina

Arquitetura de Computadores II – 1COP0012

Atividades Práticas de Laboratório n. 3

Data de entrega: 15/03/2022

1) O código abaixo realiza a leitura de 2 strings (**string1 e string2**) de tamanho máximo de 100 caracteres. Complete o código desenvolvendo a função i**ntercala** (intercala o conteúdo da string 1 com o conteúdo da string 2 e armazena o resultado em string3).

Edit | Execute

Intercala strings*

```
1  .data
2  ent1: .asciiz "Insira a string 1: "
3  ent2: .asciiz "Insira a string 2: "
4  str1: .space 100
5  str2: .space 100
6  str3: .space 200
7
8  .text
9  main: la $a0, ent1 # Parâmetro: mensagem
10       la $a1, str1 # Parâmetro: endereço da string
11       jal leitura # leitura (mensagem, string)
12       la $a0, ent2 # Parâmetro: mensagem
13       la $a1, str2 # Parâmetro: endereço da string
14       jal leitura # leitura (mensagem, string)
15       la $a0, str1 # Parâmetro: endereço da string 1
16       la $a1, str2 # Parâmetro: endereço da string 2
17       la $a2, str3 # Parâmetro: endereço da string 3
18       jal intercala # intercala (str1, str2, str3)
19       move $a0,$v0 # move o retorno da string resultante
20       li $v0,4 # Código de impressão de string
21       syscall # Imprime a string intercalada
22       li $v0, 10 # Código para finalizar o programa
23       syscall # Finaliza o programa
24
25 leitura:
26
27     li $v0, 4 # Código de impressão de string
28     syscall # Imprime a string
29     move $a0, $a1 # Endereço da string para leitura
30     li $a1, 100 # Número máximo de caracteres
31     li $v0, 8 # Código de leitura da string
32     syscall # Faz a leitura da string
33     jr $ra # Retorna para a main
34
35 intercala:
```

2) Elaborar um programa, em código MIPS, que faça a leitura de uma string ASCII e verifique se a mesma é um palíndromo (retorne 1 se for palíndromo e 0 se não for palíndromo).

Universidade Estadual de Londrina

Arquitetura de Computadores II – 1COP0012

Atividades Práticas de Laboratório n. 4

Data de entrega: 05/04/2022

Exemplo de um programa que faz a leitura e escrita de uma matriz de inteiros de ordem 3x3.

```
1  .data
2  Mat: .space 36 # 3x3 * 4 (inteiro)
3  Ent1: .asciiz " Insira o valor de Mat["
4  Ent2: .asciiz "]["
5  Ent3: .asciiz "]: "
6
7  .text
8  main: la $a0, Mat # Endereço base de Mat
9        li $a1, 3 # Número de linhas
10       li $a2, 3 # Número de colunas
11       jal leitura # leitura(mat, nlin, ncol)
12       move $a0, $v0 # Endereço da matriz lida
13       jal escrita # escrita(mat, nlin, ncol)
14       li $v0, 10 # Código para finalizar o programa
15       syscall # Finaliza o programa
16
17 indice:
18    mul $v0, $t0, $a2 # i * ncol
19    add $v0, $v0, $t1 # (i * ncol) + j
20    sll $v0, $v0, 2 # [(i * ncol) + j] * 4 (inteiro)
21    add $v0, $v0, $a3 # Soma o endereço base de mat
22    jr $ra # Retorna para o caller
23
24 leitura:
25    subi $sp, $sp, 4 # Espaço para 1 item na pilha
26    sw $ra, ($sp) # Salva o retorno para a main
27    move $a3, $a0 # aux = endereço base de mat
28 l: la $a0, Ent1 # Carrega o endereço da string
29    li $v0, 4 # Código de impressão de string
30    syscall # Imprime a string
31    move $a0, $t0 # Valor de i para impressão
32    li $v0, 1 # Código de impressão de inteiro
33    syscall # Imprime i
34    la $a0, Ent2 # Carrega o endereço da string
35    li $v0, 4 # Código de impressão de string
36    syscall # Imprime a string
37    move $a0, $t1 # Valor de j para impressão
38    li $v0, 1 # Código de impressão de inteiro
39    syscall # Imprime j
40    la $a0, Ent3 # Carrega o endereço da string
```

```
41      li $v0, 4 # Código de impressão de string
42      syscall # Imprime a string
43      li $v0, 5 # Código de leitura de inteiro
44      syscall # Leitura do valor (retorna em $v0)
45      move $t2, $v0 # aux = valor lido
46      jal indice # Calcula o endereço de mat[i][j]
47      sw $t2, ($v0) # mat[i][j] = aux
48      addi $tl, $tl, 1 # j++
49      blt $tl, $a2, l # if(j < ncol) goto l
50      li $tl, 0 # j = 0
51      addi $t0, $t0, 1 # i++
52      blt $t0, $al, l # if(i < nlin) goto l
53      li $t0, 0 # i = 0
54      lw $ra, ($sp) # Recupera o retorno para a main
55      addi $sp, $sp, 4 # Libera o espaço na pilha
56      move $v0, $a3 # Endereço base da matriz para retorno
57      jr $ra # Retorna para a main
58
59  escrita:
60      subi $sp, $sp, 4 # Espaço para 1 item na pilha
61      sw $ra, ($sp) # Salva o retorno para a main
62      move $a3, $a0 # aux = endereço base de mat
63  e:  jal indice # Calcula o endereço de mat[i][j]
64      lw $a0, ($v0) # Valor em mat[i][j]
65      li $v0, 1 # Código de impressão de inteiro
66      syscall # Imprime mat[i][j]
67      la $a0, 32 # Código ASCII para espaço
68      li $v0, 11 # Código de impressão de caractere
69      syscall # Imprime o espaço
70      addi $tl, $tl, 1 # j++
71      blt $tl, $a2, e # if(j < ncol) goto e
72      la $a0, 10 # Código ASCII para newline ('\n')
73      syscall # Pula a linha
74      li $tl, 0 # j = 0
75      addi $t0, $t0, 1 # i++
76      blt $t0, $al, e # if(i < nlin) goto e
77      li $t0, 0 # i = 0
78      lw $ra, ($sp) # Recupera o retorno para a main
79      addi $sp, $sp, 4 # Libera o espaço na pilha
80      move $v0, $a3 # Endereço base da matriz para retorno
81      jr $ra # Retorna para a main
```

EXERCÍCIOS

1) Elaborar um programa, em código MIPS, que faça a leitura de uma matriz A de inteiros, de ordem 4x3, e a leitura de um vetor de inteiros V com 3 elementos. Determinar o produto de A por V.

2) Dizemos que uma matriz inteira Anxn é uma matriz de permutação se em cada linha e em cada coluna houver n-1 elementos nulos e um único elemento igual a 1.

Exemplo:

A matriz abaixo é de permutação:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Dada uma matriz inteira Anxn, elaborar um programa, em código MIPS, para verificar se a matriz A é de permutação.

3) Dada uma matriz de inteiros Amxn, imprimir o número de linhas e o número de colunas nulas da matriz.

Exemplo: m = 4 e n = 4

$$\begin{pmatrix} 1 & 0 & 2 & 3 \\ 4 & 0 & 5 & 6 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

tem 2 linhas nulas e 1 coluna nula.

Universidade Estadual de Londrina

Arquitetura de Computadores II – 1COP0012

Atividades Práticas de Laboratório n. 5

Data de entrega: 20/04/2022

1) Elaborar um programa, em código MIPS, que faça a leitura de uma matriz de caracteres de ordem N (com N limitado a 8) e apresente como saída a matriz codificada pelo código de **César de ordem 3**.

2) Elaborar um programa, em código MIPS, que realize a leitura de duas matrizes de números inteiro de ordem N (matriz quadrada, com N limitado a 6) e apresente como resposta:

- quantos valores iguais estão na mesma posição em ambas as matrizes;
- a soma das posições (linha+coluna) de todos os elementos iguais que estão na mesma posição em ambas as matrizes.

3) Elaborar um programa, em código MIPS, que faça a leitura de uma matriz de números inteiros quadrada de ordem N (com N limitado a 8) e apresente como saída:

- o resultado da subtração: da somatória dos elementos acima da diagonal superior com a somatória dos elementos abaixo da diagonal principal;
- o maior elemento acima da diagonal principal;
- o menor elemento abaixo da diagonal principal;
- a matriz ordenada (ordem crescente).

Questão para ser entregue hoje:

Elaborar um programa em código MIPS que realize a leitura de um vetor de N (sendo N par) elementos inteiros (utilizando alocação dinâmica) e imprima como resultado o vetor que apresente nas N/2 primeiras posições os menores elementos ordenados em ordem crescente e nas N/2 posições restantes os maiores elementos ordenados em ordem decrescente.