

## Trabalho Prático 7

### Processamento de *Arrays* em *Assembly*

#### Objetivos

- Utilizar o segmento de dados do MIPS.
- Compreender o modo de organização da memória em *Arrays*.

#### Introdução

Um *array* é uma estrutura de dados usada para armazenar grandes quantidades de dados do mesmo tipo. Os vários elementos do *array* ocupam posições de memória contiguas e cada um deles pode ser acessado a partir do endereço base do *array* e do seu índice respetivo. O endereço do elemento  $i$  de um *array*,  $\&array[i]$ , é obtido através da soma do endereço base,  $array$ , com o produto do índice  $i$  pelo tamanho dos elementos contidos no *array*.

$$\text{endereço\_de\_array}[i] = \text{array} + i * \text{tamanho\_de\_cada\_elemento}$$

#### Guião

1. Considere o pedaço de código abaixo em *Assembly* do MIPS:

```

        .data
a:      .word    0xffffffffa5
b:      .word    0x000003ab
c:      .space   4

        .text
        .globl  main
main:
        . . .
la      $a0, a
la      $a1, b
la      $a3, c
lw      $t0, 0($a0)
lw      $t1, 0($a1)
sll    $t1, $t1, 2
sub     $t2, $t0, $t1
sw      $t2, 0($a3)
        . . .

```

- Indique que valor é armazenado na variável  $c$ . Exprima-o em hexadecimal.
- Converta o valor anterior para decimal na forma 'sinal + valor absoluto' da quantidade.

2. Considere o seguinte programa que converte uma *string* de caracteres de letras minúsculas numa *string* de caracteres de letras maiúsculas:

```
void main(void)
{
    static char minus[] = "texto em minusculas" ;
    static char maius[20];
    int i=0;

    while( minus[i] != '\0')
    {
        maius[i] = minus[i] + 'A' - 'a';
        i++;
    }
    printstr( maius );
}

```

- a) Codifique o programa em *Assembly*, e teste o seu funcionamento alterando os valores iniciais da *string* `minus`.
- b) Sugira uma alteração ao código fornecido de modo a que ele apresente um funcionamento mais robusto.
- c) A partir do programa obtido na alínea anterior escreva um novo programa que converta maiúsculas em minúsculas.

3. O programa seguinte lê do teclado uma *string*, conta o número de caracteres numéricos que ela contém e imprime esse resultado:

```
void main(void)
{
    static char prompt1[] = "Introduza uma string\n";
    static char result[] = "O número de caracteres numéricos: ";
    static char str[40];
    int i,n

    print_str( prompt1 );
    read_string( str, 40 );

    n=0;
    for ( i = 0; str[i]!='\0'; i++)
    {
        if (str[i] >= '0' ) && (str[i] <= '9' )
            n++;
    }

    print_str( result );
    print_int( n );
}

```

## Exercícios Adicionais

Os exercícios seguintes referem-se à manipulação de *arrays* de inteiros. Note que agora o tamanho de cada um dos elementos do *array* passa a ser 4 bytes.

1. O programa seguinte, escrito em C, imprime na consola – usando *system calls* – o conteúdo de um *array* de inteiros previamente inicializado.

```
void main(void)
{
    static int lista[] = {4, 3 , -2, 1, 27, 45};
    int i;

    print_str("O conteudo do Array é: \n");
    for (i=0; i < 6; i ++)
    {
        print_int10(lista[i]);
        print_str(" - ");
    }
}
```

Traduza o programa para *Assembly* e teste o seu funcionamento.

2. O programa seguinte lê, do teclado, um conjunto de 6 números inteiros e armazena-os no *array* *lista*.

```
void main(void)
{
    static int lista[6];
    int i=0;
    printstr( "Insira 6 numeros: ");

    for( i=0; i<6; i++)
    {
        lista[i] = read_int();
    }
}
```

a) Traduza o programa para *Assembly* e verifique o seu funcionamento usando a janela do segmento de dados do simulador MARS.

b) Altere o programa anterior (usando o programa da questão 1) de modo a que imprima na consola os números inteiros lidos.

3. O seguinte programa ordena o *array* de inteiros lido, antes de o imprimir.

```
#define SIZE 6
void main(void)
{
    static int lista[SIZE];
    int houveTroca;
    int aux,i;
    // inserir código para leitura de valores e
    // preenchimento do array
    do {
        houveTroca = FALSE;
        for (i = 0; i < SIZE-1; i++ )
        {
```

```
        if ( lista[i] > lista[i+1] )
        {
            aux = lista[i];
            lista[i] = lista[i+1];
            lista[i+1] = aux;
            houveTroca = TRUE;
        }
    }
} while (houveTroca == TRUE);
// inserir código de impressão do conteúdo do array
}
```