

## Praticas 07c

O objetivo do programa eh comparar a igualdade entre 2 vetores de inteiros de mesmo tamanho n, para testar a construção de rotinas *call/ret*. O programa usa rotinas para ler o tamanho, para ler os elementos do vetor e para comparar.

Observe que a chamada de uma rotina é feita com a instrução *call*, que empilha o endereço de retorno (endereço seguinte da instrução *call*) na pilha do sistema e ao final da rotina, uma instrução *ret* é usada para desempilhar o referido endereço e retornar a execução para o endereço a instrução *call*.

Cuidado ao utilizar a pilha diretamente ou indiretamente por meio de funções de bibliotecas. Lembre-se de deixa-la no estado correto para que o retorno funcione. Observe também que uma rotina exige uma convenção de como os parâmetros são passados, podendo ser passados pela pilha ou por registradores.

Vamos experimentar a função de biblioteca C, *getchar()*, que faz a leitura de um caracter. O caracter lido é retornado no registrador *%eax*.

Para gerar o executavel, gere primeiro o objeto executando o seguinte comando:

```
as praticas_07c.s -o praticas_07c.o
```

e depois link dinamicamente com o seguinte comando:

```
ld praticas_07c.o -l c -dynamic-linker /lib/ld-linux.so.2 -o praticas_07c
```

O executavel se chamara *praticas\_08a*, sem extensão, e para executá-lo digite:

```
./praticas_07c
```

```
.section .data
```

```
apresenta: .asciz    "\n*** Programa Compara Igualdade entre Vetores 1.0\n***\n\n"
```

```
pedetam:   .asciz    "Digite o tamanho do vetor (0 < tam <= 50) => "
```

```
pedenum:   .asciz    "Entre com o numero %d => "
```

```
info1:     .asciz    "\nLEITURA DO VETOR %d:\n\n"
```

```
info2:     .asciz    "\nVetor %d Lido : "
```

```
info3:     .asciz    "\nComparando ...\n"
```

```
formain:   .asciz    "%d"
```

```
formaout:  .asciz    " %d"
```

```
respigual: .asciz    "Vetores Iguais!\n\n"
```

```
respdifer: .asciz    "Vetores Diferentes!\n\n"
```

```

pulalinha: .asciz    "\n"

pergcont:  .asciz    "\nDeseja nova execucao <s>im ou <n>ao? => "

limpabuf:  .string   "%*c"


maxtam:    .int      50

tam:       .int      0

n:         .int      0

num:       .int      0

resp:      .int      0

vetor1:    .space    204          # 4 bytes para cada numero a ser armazenado

vetor2:    .space    204          # 4 bytes para cada numero a ser armazenado

.section .text

```

A seguir uma rotina para ler o tamanho do vetor e checar os limites permitidos. O numero deve ser maior que zero e menor que maxtam. O valor lido é retornado no registrador %ecx.

```

letam:
    pushl   $tam
    pushl   $formain
    call    scanf
    pushl   $pulalinha
    call    printf
    addl    $12, %esp # desfaz os ultimos 3 push's
    movl    tam, %ecx
    cmpl    $0, %ecx
    jle     letam
    cmpl    maxtam, %ecx
    jg      letam
    ret

```

A seguir uma rotina para ler os numeros do vetor. O endereco do vetor deve estar em %edi e o tamanho em %ecx.

```

levet:
    movl    $0, %ebx
volta1:
    incl    %ebx
    pushl   %edi
    pushl   %ecx
    pushl   %ebx
    pushl   $pedenum
    call    printf
    pushl   $num
    pushl   $formain
    call    scanf

```

```

    addl    $12, %esp # desfaz os ultimos 4 push's
    popl    %ebx
    popl    %ecx
    popl    %edi
    movl    num, %eax
    movl    %eax, (%edi)
    addl    $4, %edi
    loop    volta1
    ret

```

Segue uma rotina para mostrar os numeros do vetor. O endereco do vetor deve estar em %edi e o tamanho em %ecx.

mostravet:

```

    pushl    %edi
    pushl    %ecx
    movl    (%edi), %eax
    pushl    %eax
    pushl    $formaout
    call     printf
    addl    $8, %esp
    popl    %ecx
    popl    %edi
    addl    $4, %edi
    loop     mostravet
    pushl    $pulalinha
    call     printf
    addl    $4, %esp
    ret

```

Segue uma rotina que compara 2 strings, cujos enderecos devem estar nos registradores %edi e %esi e o tamanho em %ecx

comparastr:

```

    movl    (%edi), %eax
    movl    (%esi), %ebx
    cmpl    %eax, %ebx
    jnz     acabou
    addl    $4, %edi
    addl    $4, %esi
    loop     comparastr
    cmpl    %eax, %eax

```

acabou:

```

    ret

```

.globl \_start

\_start:

```

    pushl    $apresenta
    call     printf
    addl    $4, %esp

```

le\_n:

```

    pushl    $pedetam
    call     printf
    addl    $4, %esp

```

```
    call    letam
    movl    %ecx, n
```

le\_vetores:

```
    pushl   $1
    pushl   $info1
    call    printf
    addl    $8, %esp
    movl    $vetor1, %edi
    movl    n, %ecx
    call    levet
```

```
    pushl   $2
    pushl   $info1
    call    printf
    addl    $8, %esp
    movl    $vetor2, %edi
    movl    n, %ecx
    call    levet
```

mostra\_vetores:

```
    pushl   $1
    pushl   $info2
    call    printf
    addl    $8, %esp
    movl    $vetor1, %edi
    movl    n, %ecx
    call    mostravet
```

```
    pushl   $2
    pushl   $info2
    call    printf
    addl    $8, %esp
    movl    $vetor2, %edi
    movl    n, %ecx
    call    mostravet
```

compara\_vetores:

```
    pushl   $info3
    call    printf
    addl    $4, %esp

    movl    n, %ecx
    movl    $vetor1, %edi
    movl    $vetor2, %esi
    call    comparastr
    jz      saoiguais
    pushl   $respdifer
    call    printf
    addl    $4, %esp
    jmp     fim
```

saoiguais:

```
    pushl   $respigual
```

```

        call    printf
        addl    $4, %esp

fim:
        pushl   $pergcont
        call    printf
        pushl   $limpabuf
        call    scanf
        addl    $8, %esp
        call    getchar
        cmpl    '$s', %eax
        jz      _start
        pushl   $0
        call    exit

```

**DESAFIO:** Altere o programa para comparar se os vetores possuem exatamente os mesmos elementos, independente de posição. **MÉTODO:** percorra todos os elementos do vetor1 e para cada elemento tente localizá-lo no vetor2. Se ele não for encontrado a resposta será **VETORES NÃO POSSUEM OS MESMOS ELEMENTOS**. Se ele for encontrado, retire-o do vetor2, sobrescrevendo o último elemento do vetor2 na posição do elemento encontrado e reduzindo o tamanho do vetor. Repita o processo. Use variáveis diferentes para controlar o tamanho de cada vetor, tipo n1 e n2, as quais inicialmente são iguais e n2 vai diminuindo a cada elemento encontrado. Se todos os elementos do vetor1 forem encontrados e o n2 ficou com 0 (zero) então a resposta será **VETORES POSSUEM OS MESMOS ELEMENTOS**. Este desafio é para entregar valendo ajuda.