

Praticas 06b

O objetivo do programa é ordenar 3 números inteiros lidos nas variáveis n1, n2 e n3. A técnica utilizada foi transferir tais valores para os registradores %eax, %ebx e %ecx, depois compará-los e trocá-los entre si usando a instrução xchgl.

Ao invés de utilizar um número de resposta, a reexecução do programa foi verificada com a leitura de uma resposta na forma de caractere ('s' ou 'S'). Foi utilizada a função da biblioteca c chamada getchar() que lê exatamente um caractere do buffer do teclado e retorna no registrador %eax.

Entretanto, em programas que utilizam a função scanf(), o buffer do teclado pode ainda conter resíduos anteriormente digitados, tipo o caractere "newline" ou outros caracteres que sobraram da leitura anterior, pois a função scanf() não esvazia o buffer do teclado, apenas coleta os bytes necessários para preencher o tipo do dado requisitado e deixa o restante no buffer do teclado, conforme já explicado anteriormente.

Assim, uma leitura de caractere com getchar() somente funcionará corretamente se o buffer estiver vazio, caso contrário, a função já retornará o primeiro caractere de resíduo do buffer e nem esperará o usuário digitar o caractere que deseja.

Assim, torna-se necessário garantir que o buffer esteja vazio, para isso, o programa deve ler todos os caracteres do buffer do teclado, descartando-os. Em linguagem C, isso é feito com o próprio scanf(), passando a string de formatação "%*c", que informa para ler todos os caracteres do buffer.

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

```
as praticas_06b.s -o praticas_06b.o
```

e depois link dinamicamente com o seguinte comando:

```
ld praticas_06b.o -l c -dynamic-linker /lib/ld-linux.so.2 -o praticas_06b
```

O executável se chamara praticas_06c, sem extensão, e para executá-lo digite:

```
./praticas_06b
```

```
.section .data
```

```
abertura:      .asciz      "\n\nPrograma de Ordenacao de 3 Inteiros\n\n"
pergunta:      .asciz      "\n\nDeseja Executar de Novo? <S>IM ou <N>A0? "
tchau:         .asciz      "\n\ntchau tchau!\n\n"
pede_n1:       .asciz      "\nDigite n1 => "
pede_n2:       .asciz      "\nDigite n2 => "
pede_n3:       .asciz      "\nDigite n3 => "
mostra_orig:   .asciz      "\nOrdem Original: %d -> %d -> %d\n"
mostra_ord:    .asciz      "\nOrdem Alterada: %d -> %d -> %d\n"
```

```
n1:            .int        0
n2:            .int        0
```

```

n3:          .int      0

resp:        .int      0

formato:     .asciz    "%d"

limpabuf:    .string   "%*c"    # ou .asciz "%*c"

.section     .text

.globl      _start
_start:

volta:
    pushl    $abertura
    call     printf

    pushl    $pede_n1
    call     printf

    pushl    $n1
    pushl    $formato
    call     scanf

    pushl    $pede_n2
    call     printf

    pushl    $n2
    pushl    $formato
    call     scanf

    pushl    $pede_n3
    call     printf

    pushl    $n3
    pushl    $formato
    call     scanf

    pushl    n3
    pushl    n2
    pushl    n1
    pushl    $mostra_orig
    call     printf

    addl     $56, %esp

    movl     n1, %eax
    movl     n2, %ebx
    movl     n3, %ecx

    cmpl     %eax, %ebx
    jl       troca1

continua1:

    cmpl     %ebx, %ecx
    jl       troca2

```

continua2:

```
    cml    %eax, %ebx
    jge    fim
    xchgl  %ebx, %eax
    jmp    fim
```

troca1:

```
    xchgl  %eax, %ebx
    jmp    continua1
```

troca2:

```
    xchgl  %ebx, %ecx
    jmp    continua2
```

fim:

```
    pushl  %ecx
    pushl  %ebx
    pushl  %eax
    pushl  $mostra_ord
    call   printf

    pushl  $pergunta
    call   printf

    pushl  $resp
    pushl  $limpabuf
    call   scanf
    call   getchar      #retorna no %eax

    addl   $32, %esp
    cml    '$s', %eax
    jz     volta

    cml    '$S', %eax
    jz     volta

    pushl  $tchau
    call   printf
    addl   $4, %esp

    pushl  $0
    call   exit
```

DESAFIO: Ordene 4.