

Praticas 08a

O objetivo do programa eh aproveitar as práticas anteriores com vetores para implementar o algoritmo de ordenação por Seleção Direta. Nesse método, o qual é o menos eficiente de todos, o algoritmo busca o menor elemento dentre todos do vetor, e o insere no início, trocando com o primeiro elemento. Então, repete a operação para o restante do vetor, excetuando o primeiro elemento que já está ordenado.

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

```
as praticas_08a.s -o praticas_08a.o
```

e depois link dinamicamente com o seguinte comando:

```
ld praticas_08a.o -l c -dynamic-linker /lib/ld-linux.so.2 -o praticas_08a
```

O executavel se chamara praticas_08a, sem extensão, e para executá-lo digite:

```
./praticas_08a
```

```
.section .data
```

```
titulo: .asciz    "\n*** Programa Ordena Vetor 1.0 ***\n\n"
```

```
pedetam: .asciz    "Digite o tamanho do vetor (maximo=20) => "
```

```
formato: .asciz    "%d"
```

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

```
mostra1: .asciz    "Elementos Lidos:"
```

```
mostra2: .asciz    " %d"
```

```
mostra3: .asciz    "\nElementos Ordenados:"
```

```
pulalin: .asciz    "\n"
```

```
maxtam: .int      20
```

```
tam: .int  0
```

```
num: .int  0
```

```
soma: .int  0
```

```
vetor: .space  80
```

```
.section .text
```

```
.globl _start
```

```
_start:
```

```
    pushl $titulo
    call  printf
```

letam:

```
    pushl $pedetam
    call  printf
    pushl $tam
    pushl $formato
    call  scanf
    pushl $pulalin
    call  printf

    movl  tam, %ecx
    cmpl  $0, %ecx
    jle   letam
    cmpl  maxtam, %ecx
    jg    letam

    movl  $vetor,%edi
    addl  $16, %esp
    movl  $0, %ebx
```

lenum:

```
    incl  %ebx
    pushl %edi
    pushl %ecx
    pushl %ebx

    pushl $pedenum
    call  printf
    pushl $num
    pushl $formato
    call  scanf
    pushl $pulalin
    call  printf
    addl  $16, %esp

    popl  %ebx
    popl  %ecx
    popl  %edi
    movl  num, %eax
    movl  %eax, (%edi)
    addl  $4, %edi
    loop  lenum
```

mostravet:

```
    pushl $mostra1
    call  printf
    addl  $4, %esp
    movl  tam, %ecx
    movl  $vetor, %edi
```

mostranum:

```
    movl  (%edi), %ebx
    addl  $4, %edi
```

```

pushl %edi
pushl %ecx
pushl %ebx

pushl $mostra2
call printf
addl $8, %esp

popl %ecx
popl %edi
loop mostranum

movl tam, %ecx
cmpl $1, %ecx
jle mostravet2 # nao tem mais oque fazer

```

ordenavetor:

```

movl $vetor, %edi # inicia a posicao do primeiro

movl %edi, %edx # inicia a posicao do menor
movl %edi, %esi # inicia a posicao do proximo a ser comparado
subl $1, %ecx
pushl %ecx # backup do nro de elementos a comparar

```

giro:

```

addl $4, %esi # avanca para o proximo
movl (%edx), %eax # contem o menor valor ateh entao

```

identificado

```

movl (%esi), %ebx # contem o valor do proximo
cmpl %eax, %ebx

```

```

jl trocaposicao

```

volta:

```

loop giro

```

trocaelem:

```

movl (%edi), %ebx
movl (%edx), %eax
movl %eax, (%edi)
movl %ebx, (%edx)
addl $4, %edi # avanca o primeiro
movl %edi, %edx # inicia a posicao do menor
movl %edi, %esi # inicia a posicao do proximo a ser comparado

```

```

popl %ecx
subl $1, %ecx
pushl %ecx # backup do nro de elementos a comparar

```

```

cmpl $0, %ecx
jle mostravet2 # nao tem mais oque fazer

```

```

jmp giro

```

trocaposicao:

```

movl %esi, %edx # troca a posicao do menor
jmp volta

```

```

mostravet2:
    pushl $mostra3
    call printf
    addl $4, %esp
    movl tam, %ecx
    movl $vetor, %edi

mostranum2:
    movl (%edi), %ebx
    addl $4, %edi

    pushl %edi
    pushl %ecx
    pushl %ebx

    pushl $mostra2
    call printf
    addl $8, %esp

    popl %ecx
    popl %edi
    loop mostranum2

    pushl $pulalin
    call printf
    pushl $pulalin
    call printf
    addl $8, %esp
fim:
    pushl $0
    call exit

```

DESAFIO: Implemente o algoritmo de ordenação "Inserção Direta".