**Instituto de Computação - Unicamp**

**MC102 - Algoritmos e Programação de Computadores**

# Laboratório 14 - Sistema da DAC

Prazo de entrega: **19/05/2017 23:59:59**

Peso: **1**

*Professor:* Eduardo C. Xavier
*Professor:* Guido Araújo

*Monitor:* Arthur Pratti Dadalto
*Monitor:* Cristina Cavalcante
*Monitor:* Klairton de Lima Brito
*Monitor:* Luís Felipe Mattos
*Monitor:* Paulo Finardi
*Monitor:* Paulo Lucas Rodrigues Lacerda
*Monitor:* Pedro Alves
*Monitor:* Renan Vilas Novas
*Monitor:* Vinicius de Novaes Guimarães Pereira

## Descrição

O sistema de gerenciamento de turmas da DAC deve ser capaz de realizar operações como impressão, ordenação, inclusão, remoção e busca de alunos matriculados em cada turma.

## Objetivo

Você deve criar um programa, que dada uma lista inicial de RAs de alunos

matriculados em uma turma, consiga realizar as seguintes operações:

- p - Imprimir

A impressão deve imprimir a lista de RAs de alunos matriculados em seu estado atual, ou seja, caso a lista não tenha sido ordenada ainda, deve imprimir na mesma ordem em que foi lida na entrada. Caso a turma não tenha nenhum aluno matriculado, nada deve ser impresso.

- c - Ordenação crescente

Deve-se ordernar os RAs dos alunos em ordem crescente. Se na entrada tivermos os valores

| 190187 | 189619 | 192089 | 163465 | 192924 | 168725 | 191170 |
|---|---|---|---|---|---|---|

então a saída deve ser:

| 163465 | 168725 | 189619 | 190187 | 191170 | 192089 | 192924 |
|---|---|---|---|---|---|---|

- d - Ordenação decrescente

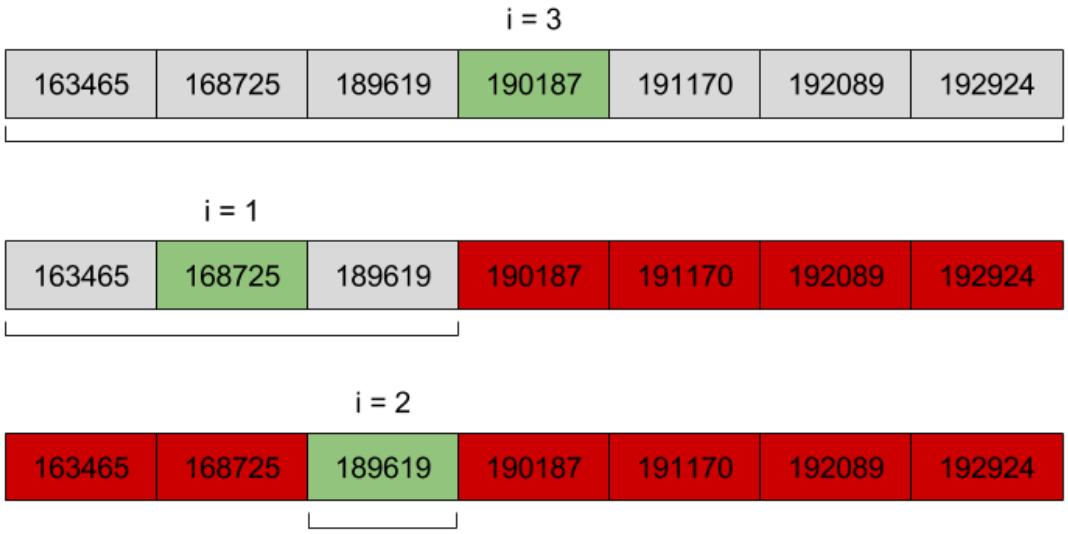
Deve-se ordenar os RAs dos alunos em ordem decrescente. Para os RAs do exemplo anterior, deve ser impresso:

| 192924 | 192089 | 191170 | 190187 | 189619 | 168725 | 163465 |
|---|---|---|---|---|---|---|

- b - Busca binária

Deve-se realizar a busca binária para um RA fornecido. A busca binária deve imprimir o índice de busca para cada passo do algoritmo e o índice final, caso o RA tenha sido encontrado ou uma mensagem de erro, caso contrário. Se a lista não estiver ordenada ao realizar a busca, uma

mensagem de erro deve ser impressa. Por exemplo, para a busca do RA 189619 na lista em ordem crescente deveria ser impresso os indices 3, 1, e 2.




A busca binária deve ser feita tanto se a lista estiver em ordem crescente quanto decrescente.

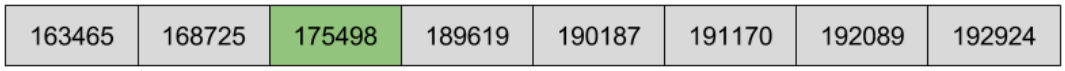
OBS: Durante a busca binária em um sub-vetor que começa na posição `i` e termina em `f`, a posição do meio é dada por `(i+f)/2`.

- i - Inserir

A inserção deve ser feita de modo que a ordenação seja mantida, ou seja, se a lista tiver sido ordenada antes da inserção, o valor inserido deve ficar na posição correta e a lista deve ser atualizada. Se a lista não tiver sido ordenada antes da inserção, o aluno é inserido ao final da lista. Caso o limite máximo de alunos seja atingido (150 alunos), uma mensagem de erro deve ser impressa ao invés de inserir o aluno na turma. Caso o aluno a ser inserido já esteja matriculado na turma, uma mensagem deve ser impressa.

Por exemplo, ao inserir o RA 175498 se o vetor não estiver ordenado


Se estiver em ordem crescente




Se estiver em ordem decrescente


- r - Remover

A remoção não pode ser realizada se não tiver alunos matriculados na disciplina, assim, uma mensagem de erro deve ser impressa. Caso o aluno não esteja matriculado na disciplina, também deve-se imprimir uma mensagem de erro.

Por exemplo, ao remover o RA 189619 da lista


obetemos a lista


- s - Sair

Finaliza a execução do programa.

Os parâmetros de cada identificador são informados imediatamente depois do mesmo, sendo que os parâmetros são:

- O identificador `b` é seguido de um número que denotamos por

`ra_busca`.

- O identificador `i` é seguido de um número que denotamos por `ra_insercao`.
- O identificador `r` é seguido de um número que denotamos por `ra_remocao`.

OBSERVAÇÕES:

- O número de cada RA é um valor do tipo inteiro.
- O número máximo de alunos que podem ser matriculados é 150.
- A busca que é feita na inserção e na remoção para verificar se o aluno pertence ou não à turma não precisa ser implementada utilizando uma busca binária, pode ser uma busca linear.
- A busca binária deve ser feita tanto quando a lista está ordenada de forma crescente, quanto quando está ordenada de forma decrescente.
- As mensagens de erros que devem ser impressas em cada caso estão listadas na seção `Saída`.

## Entrada

A entrada consiste de: um valor inteiro `n` (número de alunos a serem matriculados inicialmente na turma), uma lista com `n` inteiros (RA de cada aluno) e uma lista de operações a serem realizadas finalizada pelo caractere `s`.

## Saída

Deverá ser impressa a lista como lida na entrada e, para cada operação 'p' realizada deve ser impressa a lista no estado atual, dadas as operações realizadas anteriormente.

Quando o programa ler o operador `s`, que representa a operação de sair, o programa deve encerrar a execução.

No caso da busca binária, os índices da busca para cada passo devem ser

impressos, independente do RA estar na lista ou não. Para isso, basta imprimir o índice da posição do meio da lista durante a busca.

*Dica para a linguagem C:* imprima a saída da seguinte forma:

- Para a busca binária, quando a lista não está ordenada:

```
printf("Vetor nao ordenado!\n");
```

- Para a busca binária, quando encontra o RA desejado:

```
printf("%d esta na posicao: %d\n", ra, pos);
```

- Para a busca binária, quando o RA não está na lista:

```
printf("%d nao esta na lista!\n", ra);
```

- Para a inserção, quando o limite máximo de alunos matriculados é excedido:

```
printf("Limite de vagas excedido!\n");
```

- Para a inserção, quando o RA já está na lista:

```
printf("Aluno ja matriculado na turma!\n");
```

- Para a remoção, quando a turma não possui alunos matriculados:

```
printf("Nao ha alunos cadastrados na turma!\n");
```

- Para a remoção, quando o RA não está na lista:

```
printf("Aluno nao matriculado na turma!\n");
```

# Exemplos

Teste 01

Entrada

```
7
165291 151558 152595 173589 161817 152602 192032
p
c
p
i 182220
p
r 161817
p
b 151558
r 181322
p
s
```

Saída

```
165291 151558 152595 173589 161817 152602 192032
151558 152595 152602 161817 165291 173589 192032
151558 152595 152602 161817 165291 173589 182220 192032
151558 152595 152602 165291 173589 182220 192032
3 1 0
151558 esta na posicao: 0
Aluno nao matriculado na turma!
151558 152595 152602 165291 173589 182220 192032
```

Teste 02

Entrada

```
10
178816 150656 156803 189094 181654 175495 191114 170387 188941 163086
p
r 181654
```

```
p
c
p
i 160478
p
i 170387
p
d
p
r 199113
p
s
```

Saída

```
178816 150656 156803 189094 181654 175495 191114 170387 188941 163086
178816 150656 156803 189094 175495 191114 170387 188941 163086
150656 156803 163086 170387 175495 178816 188941 189094 191114
150656 156803 160478 163086 170387 175495 178816 188941 189094 191114
Aluno ja matriculado na turma!
150656 156803 160478 163086 170387 175495 178816 188941 189094 191114
191114 189094 188941 178816 175495 170387 163086 160478 156803 150656
Aluno nao matriculado na turma!
191114 189094 188941 178816 175495 170387 163086 160478 156803 150656
```

Teste 03

Entrada

```
5
156160 199178 153619 186388 193574
p
r 182790
p
c
p
r 199178
p
i 195262
p
```

```
i 189109
p
d
p
i 165443
p
c
p
b 153619
b 180665
i 189714
p
i 156307
p
s
```

Saída

```
156160 199178 153619 186388 193574
Aluno nao matriculado na turma!
156160 199178 153619 186388 193574
153619 156160 186388 193574 199178
153619 156160 186388 193574
153619 156160 186388 193574 195262
153619 156160 186388 189109 193574 195262
195262 193574 189109 186388 156160 153619
195262 193574 189109 186388 165443 156160 153619
153619 156160 165443 186388 189109 193574 195262
3 1 0
153619 esta na posicao: 0
3 1 2
180665 nao esta na lista!
153619 156160 165443 186388 189109 189714 193574 195262
153619 156160 156307 165443 186388 189109 189714 193574 195262
```

Para mais exemplos, consulte os testes abertos no Susy.

## Observações

- O número máximo de submissões é **10**;

- O seu programa deve estar completamente contido em um único arquivo denominado `lab14.c`.
- Para a realização dos testes do SuSy, a compilação dos programas desenvolvidos em C irá considerar o comando:
  `gcc -std=c99 -pedantic -Wall -o lab14 lab14.c`;
- Você deve incluir, no início do seu programa, uma breve descrição dos objetivos do programa, da entrada e da saída, além do seu nome e do seu RA;
- Indente corretamente o seu código e inclua comentários no decorrer do seu programa.

## Critérios importantes

Independentemente dos resultados dos testes do SuSy, o não cumprimento dos critérios abaixo implicará em nota zero nesta tarefa de laboratório.

- Para os programas em linguagem C, os headers aceitos para essa tarefa são `stdio.h` e `string.h`.