Lista Simplesmente Encadeada

Uma lista é uma estrutura de dados que visa armazenar um conjunto de dados do mesmo tipo. Uma lista encadeada é um tipo especial de lista onde cada elemento possui um endereço para o próximo elemento.

Você foi incumbido da tarefa de criar um programa que utilize uma lista encadeada para armazenar um conjunto de alunos. Cada aluno possui uma matrícula (char[10]), nome (char [40]), uma data de nascimento (struct Data vista em aula), média geral de notas (float). Sua implementação deve suportar os seguintes requisitos

- Incluir elemento na lista
    - A inclusão deve ser feita na ordem: matricula, nome, data_nascimento, nota (um dado por linha);
    - Ex:
      201001039
      Allan
      23/06/1912
      10.0
- Excluir elemento da lista
    - Passa a matrícula do elemento que será excluído
    - Ex. 201001039
- Listar os elementos da lista em ordem de inserção
    - Lista todos os elementos no formato “matricula, nome, data_nascimento, nota”, um elemento por linha
        - Ex:
          201001039, Allan, 23/06/1912, 10.00
          201001079, Edgar, 19/08/1923, 9.90

- Listar os elementos da lista em ordem inversa a de inserção
    - Mesmo formato do tópico anterior
        - Ex:
          201001079, Edgar, 19/08/1923, 9.9
          201001039, Allan, 23/06/1912, 10
- Contar o número de elementos da lista
    - Retornar uma linha com um inteiro com o tamanho da lista
- O programa deve possui um menu para que seja possível a navegar entre as operações suportadas
    - Opção 1: Incluir elemento na lista
    - Opção 2: Excluir elemento da lista
    - Opção 3: Listar todos os elementos da lista na ordem de inclusão
    - Opção 4: Listar todos os elementos da lista na ordem inversa a inclusão
    - Opção 5: Apresentar quantos elementos existem na lista
    - Opção 0: Sair do programa

- Ao fim do programa (após a opção 0) a memória alocada deverá ser desalocada, e para cada elemento desalocado o programa deverá imprimir um '-'.
- Caso o usuário tente executar uma operação que não possa ser executada, por exemplo, excluir um elemento de uma lista vazia ou mesmo imprimir uma lista vazia a mensagem “Lista Vazia!” deve ser apresentada na tela

Entrada
A entrada contém uma série de casos de teste, cada uma iniciando pela opção *I* da operação desejada.

I -> Operação desejada , onde -1 < I < 6

A operação I indica qual funcionalidade o usuário deseja. Cada funcionalidade apresenta um formato específico de entrada já apresentado acima.

Saída
A saída deverá ser impressa conforme os exemplos fornecidos abaixo. Cada saída possui uma relação com a operação solicitada, e ao final de cada uma deve haver uma quebra de linha.

Obs: Você receberá um conjunto de entradas e saídas. Lembre-se que para encaminhar os elementos a sua solução você pode utilizar os direcionadores de entrada do shell. Ex. Se seu programa se chama ‘solucao’ e seus arquivos de entrada e saída, ‘entrada1.in’ e ‘saida1.out’ respectivamente, você pode utilizar os seguintes comandos para verificar se a solução está ok

```
./solucao < entrada1.in > saidaPrograma.out
diff saidaPrograma.out saida1.out
```

Exemplo 1 (entradas em azul, saídas em vermelho)

```
1
201001039
Allan
23/06/1912
10.0
1
201001079
Edgar
19/08/1923
9.9
3
4
2
201001039
3
0
201001039, Allan, 23/6/1912, 10.00
201001079, Edgar, 19/8/1923, 9.90
201001079, Edgar, 19/8/1923, 9.90
```

201001039, Allan, 23/6/1912, 10.00
201001079, Edgar, 19/8/1923, 9.90
-

Exemplo 2 (entradas em azul, saídas em vermelho)

2
201001039
1
201001039
Allan
23/6/1912
10.0
1
201001079
Edgar
19/8/1923
9.9
3
2
201001079
2
201001039
3
1
201001001
Ada
10/12/1815
10.0
5
1
201801021
Chapatin
21/2/1929
6.5
1
201801031
Gustavo
17/3/1978
8.9
5
0
Lista Vazia!
201001039, Allan, 23/6/1912, 10.00
201001079, Edgar, 19/8/1923, 9.90

Lista Vazia!
1
3
