Atividade 2 de Estruturas de Dados I
Lista Simplesmente Encadeada

Questão única: Considere um cadastro de produtos implementado através de uma lista linear simplesmente encadeada. Considere que este cadastro **não está ordenado e NÃO pode conter produtos repetidos**. Implemente um programa (aplicação) para manipular este cadastro. O programa deverá implementar o menu abaixo e um loop para permitir que o usuário manipule o cadastro.

Menu de opções
1 – Inserir novo produto no início do cadastro
2 – Inserir novo produto no final do cadastro
3 – Remover o primeiro produto do cadastro
4 – Remover o último produto do cadastro
5 – Exibir todos os produtos do cadastro
6 – Exibir os dados de um produto do cadastro
7 – Alterar os dados de um produto do cadastro
8 – Remover um produto do cadastro
0 – Sair do programa

O programa deverá utilizar cinco classes: a classe Produto, a classe LSENode, a classe LSESemRepetidos, a classe CadastroProdutos e a classe principal (aplicação).

Sobre cada produto deverão ser mantidas as seguintes informações: código (String), descrição (String), preço (double) e estoque (int). O que diferencia um produto de outro é seu código. Após criado um produto, seu código não poderá ser alterado, ou seja, o atributo código é ***final***.

A classe CadastroProdutos deverá ter um atributo lista que será uma lista simplesmente encadeada de produtos e deverá conter os seguintes métodos:

(a) **Procedimento** para inserir um novo produto no final do cadastro. Este procedimento deverá solicitar ao usuário as informações sobre o produto, fazendo a validação dos dados informados. Criar o objeto da classe Produto e inseri-lo na lista.
(b) **Procedimento** para exibir todos os produtos cadastrados.
(c) **Procedimento** para remover um produto do cadastro. Este procedimento deverá solicitar ao usuário o código do produto a ser removido e removê-lo da lista.
(d) **Procedimento** para exibir as informações de um dado produto. Este procedimento deverá solicitar ao usuário o código do produto cujos dados desejamos exibir.
(e) **Procedimento** para atualizar o preço de um dado produto. Este procedimento deverá solicitar ao usuário o código do produto cujo preço desejamos alterar, assim como o novo preço. O procedimento deve validar os dados informados pelo usuário.
(f) **Procedimento** para atualizar o estoque de um dado produto. Este procedimento deverá solicitar ao usuário o código do produto cujo estoque desejamos alterar, assim como a quantidade a ser ACRESCIDA ao estoque. O procedimento deve validar os dados informados pelo usuário.
(g) **Procedimento** para registrar a venda de um dado produto. Este procedimento deverá solicitar ao usuário o código do produto cuja venda desejamos registrar, assim como a quantidade a ser RETIRADA do estoque. O procedimento deve validar os dados informados pelo usuário e verificar se a quantidade informada é menor ou igual ao estoque existente.

A classe Lista deverá ter os seguintes métodos:

(a) **Procedimento** para inserir um novo objeto no início da lista. Este procedimento deverá receber como parâmetro o objeto a ser inserido.
(b) **Procedimento** para inserir um novo objeto no final da lista. Este procedimento deverá receber como parâmetro o objeto a ser inserido.
(c) **Procedimento** para exibir todos os objetos cadastrados na lista.
(d) **Procedimento** para remover o objeto situado no início da lista.
(e) **Procedimento** para remover o objeto situado no final da lista.
(f) **Função** para procurar um objeto na lista e, caso encontre, retornar uma referência para o nó que o contém. Caso não encontre, deverá retornar **null**. Esta função deverá ser privada.
(g) **Função** para procurar um objeto na lista e, caso encontre, retornar uma referência para o objeto encontrado. Caso não encontre, deverá retornar **null**.
(h) **Procedimento** para remover um objeto da lista. Este procedimento deverá procurar pelo objeto e, caso encontre, removê-lo. **OBS**: **NÃO** utilize as funções definidas nos itens (f) e (g).