Exercício: Depósito de Caixas (Com Condição de Corrida)

Depósito de Caixas (Com Condição de Corrida)

Assuma que um produtor vai ao depósito armazenar as caixas que vai produzindo. Da mesma forma, um consumidor vai ao depósito para retirar caixas que vai consumir.

As seguintes atividades devem ser feitas:

1) Implemente uma classe Deposito que tenha os atributos: (i) a quantidade itens atualmente no depósito (zero se não for informado); e (ii) a capacidade máxima de itens no depósito (**tem que ter esse parâmetro** no construtor). E que tenha os seguintes métodos: (i) armazenar, que coloca uma caixa no depósito; e (ii) retirar, que retira uma caixa do depósito. Obviamente que não é possível armazenar nenhuma caixa caso a capacidade máxima já tenha sido atingida. Da mesma forma, não é possível retirar nenhuma caixa, caso o depósito esteja vazio.

2) Implemente uma classe **Produtor** que deve funcionar como uma thread independente e que invoca o método "armazenar" do objeto depósito de forma a acrescentar caixas ao depósito. A classe Produtor deve receber no construtor uma referência para o objeto depósito, além de um inteiro correspondente ao tempo em milisegundos entre produções de caixas. No método run, execute **MAX** produções, onde MAX é uma constante. Defina a classe Produtor como sendo uma classe que implementa o método **Runnable**.

3) Implemente uma classe **Consumidor** que também funcione como uma thread independente e que invoca o método "retirar" do objeto depósito de forma a retirar caixas do depósito. A classe Consumidor deve receber no construtor uma referência para o mesmo objeto depósito e um inteiro correspondente ao tempo em milisegundos entre consumos de caixas. No método run, execute **MAX** consumos, onde MAX é uma constante. Defina a classe Consumidor como sendo uma sub-classe da classe **Thread**.

4) Implemente uma **estratégia** para saber se realmente uma caixa foi **adicionada** ou não no depósito e se, da mesma forma, uma caixa foi **retirada** ou não do depósito.

5) No método principal, inclua **mais de um consumidor e produtor** com diferentes tempos de produção e de consumo, de tal forma que você consiga forçar a ocorrência de uma condição de corrida. Se precisar, pode também incluir o método **Thread.sleep**.

Observações:

1) Use somente um arquivo para resolver esse problema;
2) Entregue somente os códigos Java.