Lista de Exercícios 07

Threads Java

scc0204 _ Programação Orientada a Objetos

Prof. Moacir P. Ponti Jr.

23 de maio de 2011

Considere a classe “Deposito” abaixo:

```
public class Deposito {
	private int items=0;
	private final int capacidade=10;

	public int retirar() {
		if (items>0) {
			items--;
			System.out.println("Caixa retirada: existem "+items+" caixas");
			return 1;
		}
		return 0;
	}

	public int colocar () {
		if (items<capacidade) {
			items++;
			System.out.println("Caixa armazenada: existem "+items+" caixas");
			return 1;
		}
		return 0;
	}

	public static void main(String[] args) {
		Deposito dep = new Deposito();
		//Produtor p = new Produtor(dep, 2);
		//Consumidor c = new Consumidor(dep, 1);

		// iniciar thread produtora
		//...
		// iniciar thread consumidora
		//...

		System.out.println("Fim da execucao de Deposito::main()");
	}
}
```

- **EX1** : Copie o código acima num arquivo `Deposito.java`, compile e execute para verificar se está funcionando.

- **EX2** : Crie uma classe Produtor que funcione como uma thread independente e que evoque 1000 vezes o método colocar() da classe depósito de forma a acrescentar caixas ao depósito. A classe Produtor deve receber no construtor uma referência para o objeto 'dep' onde os métodos vão ser chamados e um inteiro correspondente ao tempo em segundos entre produções de caixas. Note que há um limite de capacidade do depósito. Se o produtor estourar o limite, esse deve ficar "aguardando" a condição necessária para produzir mais caixas. Para isso você pode usar o método wait() dentro de um loop :

```
while ( condition != true )
      wait();
```

  Defina a classe Produtor como sendo uma sub-classe de Thread. Declare e instancie essa classe na função main() de Deposito, compile e execute para ver o que acontece.
- **EX3** : Crie uma classe Consumidor que funcione como uma thread independente e que evoque 1000 vezes o método retirar() da classe depósito de forma a retirar caixas do depósito. Note que há um limite de capacidade do depósito. Se o consumidor for estourar o limite, esse deve ficar "aguardando" a condição necessária para produzir mais caixas. A classe Consumidor deve receber no construtor uma referência para o objeto dep onde os métodos vão ser chamados e um inteiro correspondente ao tempo em segundos entre consumos de caixas. Defina a classe Consumidor como sendo uma classe que implementa o método Runnable. Note que agora declarar e instanciar um objeto Consumidor é um pouco diferente de um Produtor, pois Produtor herda diretamente de Thread (é uma Thread), enquanto Consumidor apenas implementa funcionalidades da interface Runnable.
- **EX4** : Execute o sistema e experimente algumas variantes como:
  - Adicionar à classe consumidor mensagens que permitam identificar o que cada objeto Consumidor está a fazer em cada momento, e em particular se está bloqueado à espera que existam caixas.
  - Instancie mais threads consumidoras ou produtoras e modifique os tempos entre produções e consumos.