Aula 3

Trabalho Autónomo

Exercício 1

Descreva as diferenças entre Barreiras e BlockingQueque?

Exercício 2

O objetivo deste exercício é implementar uma classe Barrier para concretizar barreiras. Neste tipo de sincronização, um grupo de threads bloqueia na barreira até que o número de threads bloqueadas na mesma atinja determinado valor (configurável).

- Deverá ser possível instanciar a barreira indicando o número de threads necessário para desbloquear a barreira.
- A operação bloqueante deverá obedecer à seguinte assinatura:

public synchronized void await() throws InterruptedException

- Deverá existir uma operação para saber quantas threads estão bloqueadas na barreira
- A barreira deverá ser observável, dando origem a um evento por cada thread que se regista na barreira

Escreva um pequeno programa para testar o comportamento da barreira. Por exemplo, criando 5 threads, as quais imprimem uma mensagem inicial, esperam um tempo aleatório entre 1 e 5 segundos, depois registam-se na barreira (instanciada para 5), e finalmente imprimem uma mensagem final. O efeito esperado será observar as mensagens finais a aparecerem simultaneamente.

Exercício 3

Crie uma classe ContaBancaria e um método depositar.

```
class Account {
        void deposit(int amount);    // put money into the account
        int getBalance();    // get the current balance of the account
}
```

Crie 10 Threads Cliente, cada um vai fazer depósitos contínuos de um valor aleatório (por exemplo entre 0 e 100€) até ser interrompido. Cada cliente guarda o total que ele próprio depositou.

O main, fará o seguinte:

1. Lança os 10 clientes
2. dorme 10 segundos,
3. interrompe todos os clientes,
4. espera que terminem (usando o join()),
5. imprime o saldo da conta
6. soma o total depositado por todos os clientes para confirmação.

Experimente com e sem sincronização do método deposit().