UNIVERSIDADE VILA VELHA
ESPÍRITO SANTO

Curso de Ciência da Computação

1o Trabalho Computacional - Concorrente

Data da entrega: 14/09/2017

2 componentes por grupo

Objetivo:
Pretende-se com este trabalho desenvolver um sistema concorrente com o uso de threads.

Considerações:
Observe os requisitos funcionais abaixo. **O grupo pode e deve sempre que achar necessário incrementar com mais funcionalidades.** Aplique os métodos da API de Threads, quando necessário:

- **A)** Deve-se criar uma aplicação que modele o funcionamento de sua **conta bancária**. Esta conta será o *recurso compartilhado* e deverá ser acessada/modificada por três threads: o thread **AEsperta**, o thread **AEconomica** e o thread **AGastadora**, concorrentemente.
    - **1. AGastadora:** este thread (de atitude voraz) deverá, a cada 3000 milesegundos verificar se há saldo suficiente, e retirar 10 reais da sua conta. Este thread deve disputar por dinheiro, com os demais threads, concorrentemente.
    - **2. AEsperta:** este thread será mais comedido que a anterior: somente a cada 6000 milesegundos, irá verificar o seu saldo. Mas não se engane: se houver saldo suficiente, este thread irá retirar 50 reais da sua conta. Este thread deve disputar com outros threads, concorrentemente.
    - **3. AEconomica:** de todos as threads, esta será a que mais prezará por você e suas finanças. Ela irá verificar o saldo de sua conta apenas a cada 12000 milesegundos. Se houver fundos, a thread econômica irá tentar retirar apenas 5 reais da sua conta. Este thread deve disputar com outros threads, concorrentemente.

- **B)** A classe **Conta** (*recurso compartilhado*) deverá ter pelo menos:
    - Os seguintes atributos: **número** da conta, **titular** da conta e **saldo**.
    - Defina, **construtores**, métodos **get/set** para cada um dos atributos e um método **toString**.
    - Quando necessário, implemente também algumas regras básicas de integridade (número de conta negativos, etc.).
    - Finalmente, implemente os principais métodos que vão manipular o saldo da conta: **deposito** e **saque**. Inicie o saldo (recurso compartilhado) da conta depositando uma quantia de **R$ 10.000,00**.

- DICA: Faça um esboço de um diagrama inicial de classes para lhe ajudar a pensar melhor antes de ir diretamente para o código. O diagrama ajuda você a visualizar quantas classes precisará e como elas se relacionam.

- IMPORTANTE: Sempre que um thread movimentar fundos da sua conta, o sistema deve informar não apenas qual thread efetuou a operação (saque ou depósito), mas, principalmente, qual é o seu saldo atual final (após o saque). Isso permitirá que você acompanhe a situação financeira em tempo real.

- LEMBRE-SE: Nesta aplicação, não defina prioridades (todos devem ter as mesmas chances). Além disso, não permita que haja corrupção de dados

(ou seja, cada thread deve conseguir retirar sua quantia sem ser interrompido por outro thread materialista). Use synchronized ou outro modelo, por exemplo.

- **C)** Quando a conta estiver com **saldo zero**, todos os threads deverão ser colocados em estado de espera. Ao serem colocado em espera , cada thread deverá imprimir a quantidade de saques efetuados e o valor total retirado da conta. Execute e veja o resultado

- **D)** Acrescente agora mais uma thread de nome **APatrocinadora**. Este thread deverá depositar 100 reais sempre que a conta estiver zerada. Veja que este thread será "produtora". E as demais serão "consumidoras". (necessário usar wait e notifyAll ou outro modelo, por exemplo)

- **E)** Fica a critério do grupo se vão ou não usar GUI, apesar de que é fortemente recomendado.

O que deve ser entregue:

- O material (código + documentação) do trabalho. .
    1. A documentação JavaDOC deve conter todos os comentários para entendimento do programa, bem como uma parte, logo no início da classe, as decisões tomadas, etc.
    2. Projeto contendo os Algoritmos em Java,

Avaliação:

A avaliação será composta por duas partes:

- Avaliação do material pedido no item "**O que deve ser entregue**"
- **Arguição INDIVIDUAL** dos componentes do grupo, no dia da entrega do trabalho. **Cada integrante** do grupo **deverá apresentar** alguma parte.

A NOTA SERÁ INDIVIDUAL, de acordo com os critérios apresentados pelo professor aos alunos da disciplina.

Importante:

- Após a data estabelecida o trabalho não será mais aceito.
- Não serão pontuados os grupos que deixarem de entregar algum dos itens pedidos.
- Procure ter um cuidado especial com a formatação da interação com o usuário (entrada e saída de dados)

- **Valor do Trabalho Computacional: 2.0 pontos**