	UNIVERSIDADE FEDERAL DO PARÁ **FACULDADE DE COMPUTAÇÃO** **ENGENHARIA DA COMPUTAÇÃO** **2° SEMESTRE** **DATA: / /** **ALUNO:**_____	**1ª Aval. POO** **NOTA:**

Data de Entrega: 10/01/2021

Equipes: 2 pessoas (Nó máximo)

Nota: A avaliação do trabalho será realizada da seguinte forma

$PARTE\ I$: $\mathbf{10}\ \boldsymbol{pts}\ -\ PARTE\ II$: $\mathbf{10}\ \boldsymbol{pts}$

$$\boldsymbol{NF} = \frac{\big((\boldsymbol{PARTE\ I} * \mathbf{0.7}) + (\boldsymbol{PARTE\ II} * \mathbf{0.3})\big)}{\mathbf{2}}$$

Entregáveis: Os códigos funcionando com comentários pertinentes a cada classe, método e lógica, explicando o que é feito e como foi realizado. Os *arquivos.java* devem ser zipados (arquivo.zip) separadamente (Ex. Parte I, Parte II) e enviados para o Email **marcelino.silva@gmail.com** com o título "**Trabalho Final 01 ED**".

PARTE I

1. Crie uma classe correspondente a uma Conta Bancária que contém os atributos número da conta e saldo, e com os métodos sacar e depositar que recebem um parâmetro do tipo double.

2. Crie classes correspondentes a Conta Corrente e Conta Poupança que herdam da Conta Bancaria. A primeira possui um atributo taxaDeOperacao que é descontado sempre que um saque e um depósito são feitos. A segunda possui um atributo limite que dá credito a mais para o correntista caso ele precise sacar mais que o saldo. Neste caso, o saldo pode ficar negativo desde que não ultrapasse o limite. Contudo isso não pode acontecer na classe Conta Corrente.

Boa Prova!
#Feriado

3. Crie uma classe Banco que possui um vetor de contas bancárias e implemente os métodos inserir, remover e procurarConta. O primeiro e o segundo recebem um objeto conta (que pode ser corrente ou poupança) e o insere e remove no vetor, respectivamente. O terceiro recebe um inteiro como parâmetro representando o número da conta e retorna um objeto conta bancária, caso essa conta exista no vetor, ou null, caso contrário.

4. Adicione a classe Conta Bancaria com o método transferir que recebe o parâmetro o valor (double) e um objeto conta bancaria e transfere o valor desejado da conta atual para cada conta informada. Use os métodos sacar e depositar para isso.

5. Crie uma classe Relatório, com um método mostrarDados responsável por exibir os dados da conta.

6. Crie uma classe Menu (método principal) que instancie um banco e ofereça as seguintes opções:

- Criar conta: o usuário informa se é conta poupança ou corrente e os dados da conta. O objeto correspondente é criado e inserido no banco através do método inserir. Exibir uma mensagem de sucesso.
- Selecionar conta: o usuário informa o número da conta. Se a conta existir, mostra o menu abaixo. Caso contrário, mostra mensagem de conta inexistente.
- Remover conta: o usuário informa o número da conta. Se a conta existe, então ela é excluída e uma mensagem de sucesso é informada. Caso contrário, uma mensagem de conta inexistente é informada.
- Gerar Relatório: mostra os dados de todas as contas cadastradas no banco.
- Finalizar: termina a aplicação.
- Se o usuário escolher a opção 2 mostre o seguinte menu:
    - Depositar: recebe um valor e deposita na conta.
    - Sacar: recebe um valor e tenta sacar da conta.

Boa Prova!
#Feriado

- Transferir: recebe um valor e o número de outra conta. Caso a conta exista, transfere o valor de uma conta para a outra. Caso contrário, informar mensagem de conta inexistente.
- Gerar relatório: mostra os dados da conta selecionada.
- Retornar ao menu anterior: exibe o menu anterior (opções 1 a 5).

- **PARTE II**

1. Aplique os conceitos de Orientação a Objeto em Estrutura de Dados. Crie uma **Classe Fila**, uma **Classe Pilha** e uma **Classe Lista**. Cada classe terá métodos responsáveis por **inserir**, **remover** e **alterar valores** da estrutura em questão. Por fim, crie uma classe Main com um menu que possibilite ao usuário escolher com qual das 3 estruturas deseja trabalhar e em seguida pergunte qual das operações ele desejaria aplicar.

Boa Prova!
#Feriado