Faculdade IMPACTA	**Curso de Ciência da Computação** **Programação Orientada a Objetos** **Atividade Contínua 02**

Data de Entrega: 11/04/2021 (23:59)
Grupos: máximo de 6 alunos

DESCRIÇÃO

Considere que uma agência bancária tem 5 clientes e cada cliente possui uma conta.

Cliente	Saldo em Conta Corrente
Marcos	R$ 1.000,00
Julia	R$ 250,00
João	R$ 2.500,00
Roberto	R$ 3.000,00
Janaína	R$ 4.500,00

Vamos desenvolver uma aplicação em Java para gerenciar estas contas.

A CLASSE `ContaBancaria`

Como nossa aplicação precisa lidar com diferentes contas bancárias, vamos utilizar a classe `ContaBancaria` para suportar o nome do correntista, o número da conta e o saldo, cujo código é apresentado a seguir.

```
public class ContaBancaria {
	private static int ultimoNumeroConta = 1000; // Último número de conta utilizado

	private String correntista;
	private int numeroConta; // número da conta
	private double saldo; // saldo da conta

	public ContaBancaria(String correntista, double saldo) {
		ultimoNumeroConta++;
		this.numeroConta = ultimoNumeroConta;
		this.saldo = saldo;
		this.correntista = correntista;
	}
```

```
	public void depositar(double valor) {
		this.saldo = this.saldo + valor;
	}

	public void sacar(double valor) {
		this.saldo = this.saldo - valor;
	}

	public double getSaldo() {
		return this.saldo;
	}

	public int getNumeroConta() {
		return this.numeroConta;
	}

	public String toString() {
		return "Conta de " + this.correntista + " - Saldo de R$ " + this.saldo;
	}
}
```

A CLASSE Banco

Vamos fazer uma nova classe chamada `Banco`.

Esta classe irá criar cinco instâncias da classe `ContaBancaria` e armazenar as referências em um vetor.

Vamos adicionar um método `mostrarInfo` que recebe o vetor de contas bancárias e exibe na tela os dados (correntista e saldo) de todas as contas.

Vamos também adicionar as interações do programa com o usuário. Para isso, serão criados os métodos `interacaoSacar` e `interacaoDepositar`, e implementaremos um menu de opções na aplicação.

```
import java.util.Scanner;

public class Banco {

	public static Scanner entrada;

	public static void mostrarInfo(ContaBancaria[] contas) {
		System.out.println("\nContas de todos os clientes:");
		for (int i = 0; i < contas.length; i++) {
			System.out.println("[" + i + "]" + contas[i].toString());
		}
		System.out.println("");
	}
```

```
	public static void interacaoSacar(ContaBancaria[] contas) {
		boolean clienteValido = false;
		int indiceConta = -1;
		while (!clienteValido) {
			mostrarInfo(contas);
			System.out.print("O saque será efetuado na conta de qual cliente? (0 a " +
(contas.length - 1) + "): ");
			indiceConta = entrada.nextInt();
			if (indiceConta >= 0 && indiceConta < contas.length) {
				clienteValido = true;
			} else {
				System.out.println("Índice de cliente inválido!");
			}
		}

		System.out.print("Qual o valor do saque? ");
		double saque = entrada.nextDouble();
		contas[indiceConta].sacar(saque);
		System.out.println("Saque finalizado.\n");
	}

	public static void interacaoDepositar(ContaBancaria[] contas) {
		System.out.println("Em construção...");
	}

	public static void main(String[] args) {
		ContaBancaria[] contas = new ContaBancaria[5];
		contas[0] = new ContaBancaria("Marcos", 1000.00);
		contas[1] = new ContaBancaria("Júlia", 250.00);
		contas[2] = new ContaBancaria("João", 2500.00);
		contas[3] = new ContaBancaria("Roberto", 3000.00);
		contas[4] = new ContaBancaria("Janaína", 4500.00);

		entrada = new Scanner(System.in);
		boolean sair = false;

		while (!sair) {
			System.out.println("Escolha uma operação:");
			System.out.println("(1) mostrar informações de todas as contas");
			System.out.println("(2) sacar");
			System.out.println("(3) depositar");
			System.out.println("(4) sair");
			System.out.print("Opção escolhida: ");
			int escolha = entrada.nextInt();
			System.out.println();

			switch (escolha) {
			case 1:
				mostrarInfo(contas);
				break;
```

```
                    case 2:
                          interacaoSacar(contas);
                          break;
                    case 3:
                          interacaoDepositar(contas);
                          break;
                    case 4:
                          sair = true;
                          break;
                    default:
                          System.out.println("Opção inválida!");
                    }
                    System.out.println();
              }
              System.out.println("Fim do programa!");
       }
}
```

ATIVIDADE

O objetivo é que você complemente as funcionalidades dessa aplicação fazendo os exercícios a seguir:

1. Utilize como modelo o método `interacaoSacar`, e implemente o método `interacaoDepositar`.

2. Desenvolva um método na classe `ContaBancaria` para efetuar a operação de transferência. Utilize a assinatura abaixo:

   ```
   public void transferir(double valor, ContaBancaria contaDestino) {
        // instruções do método a ser desenvolvido
   }
   ```

   Este método deverá transferir o `valor` especificado da conta da instância atual (cujos membros podem ser acessados utilizando `this`) para a conta de destino (que pode ser acessada utilizando o parâmetro `contaDestino`).

3. Desenvolva o método `interacaoTransferir`, que deverá solicitar os dados necessários para a operação de transferência e efetuá-la.

4. Altere a classe ContaBancaria, para incluir um atributo privado `senha` de 6 caracteres numéricos entre `0` e `9`. Esta senha deverá ser definida no construtor e gerada automaticamente pela classe, através de um método privado utilizando geradores aleatórios. Utilize a assinatura abaixo:

   ```
   private String criarSenha() {
        // instruções do método a ser desenvolvido
   }
   ```

   Sugestão: para geradores aleatórios, utilize a classe `Random` do pacote `java.util`.

5. O governo federal resolveu voltar com a cobrança da CPMF. Para cada saque realizado, deve-se descontar 0.25% do valor do saque do saldo restante do cliente. Os valores descontados devem ser acumulados no atributo privado `cpmf`, que deve ser incluído na classe ContaBancaria.

CRITÉRIOS DE AVALIAÇÃO

O programa entregue **será avaliado** de acordo com os seguintes itens:

- Funcionamento adequado do programa;
- Implementação correta dos métodos solicitados;
- O programa deve estar na linguagem Java;
- O quão fiel é o programa quanto à descrição do enunciado;
- Identação, comentários e legibilidade do código;
- Clareza na nomenclatura de variáveis e funções.

ATENÇÃO

- Insira no início do arquivo um comentário com os nomes dos alunos do grupo.
- O nome dos métodos devem ser idênticos ao solicitado no enunciado.
- A quantidade e tipos de parâmetros de entrada e retorno dos métodos devem ser os mesmos descritos no enunciado.
- Caso você queira, métodos adicionais podem ser implementados no seu programa.
- Não serão aceitos trabalhos entregues em atraso.
- Os trabalhos passarão por ferramenta de identificação de plágio, e caso seja identificada a cópia de trabalhos, os grupos envolvidos terão a nota zerada.

Forma de entrega

Postar o código fonte do programa desenvolvido no classroom. Deve ser entregue por apenas um aluno do grupo, mas não esqueça de indicar o nome de todos os alunos do grupo.