Extra - 04

Orientação a Objetos – DCC025

Prof. Edmar Welington Oliveira
oliveira.edmar@ufjf.edu.br

Universidade Federal de Juiz de Fora – UFJF
Departamento de Ciência da Computação - DCC

2011.3

Exercício 03

Elabore uma classe ContaBancaria, com os seguintes membros:

- ✓ atributo String cliente
- ✓ atributo int num_conta
- ✓ atributo float saldo
- ✓ método sacar (o saldo não pode ficar negativo)
- ✓ método depositar

Agora acrescente ao projeto duas classes herdadas de ContaBancaria: ContaPoupança e ContaEspecial, com as seguintes características a mais:

⇒ Classe ContaPoupança:

- ✓ atributo int dia de rendimento
- ✓ método calcularNovoSaldo, recebe a taxa de rendimento da poupança e atualiza o saldo.

⇒ Classe ContaEspecial

- ✓ atributo float limite
- ✓ redefinição do método sacar, permitindo saldo negativo até o valor do limite.

Solução – Classe Contabancaria

```
 3 public class ContaBancaria {
 4
 5     private String cliente;
 6     private int numConta;
 7     private float saldo;
 8
 9     public ContaBancaria(String cliente, int numConta, float saldo){
10         this.cliente = cliente;
11         this.numConta = numConta;
12         this.saldo = saldo;
13     }
14
15     public String getCliente() {
16         return cliente;
17     }
18
19     public int getNumConta() {
20         return numConta;
21     }
22
23     public float getSaldo() {
24         return saldo;
25     }
26
27     public void setSaldo(float saldo) {
28         this.saldo = saldo;
29     }
```

Solução – Classe Contabancaria - Continuação

```
30
31      public void sacar(float valorSaque){
32          if(this.getSaldo() > valorSaque){
33              this.setSaldo(this.getSaldo() - valorSaque);
34          }
35          else{
36              System.out.println("Saldo Insuficiente");
37          }
38      }
39
40      public void depositar(float valorDeposito){
41          this.setSaldo(this.getSaldo() + valorDeposito);
42      }
43  }
```

Solução – Classe ContaPoupanca

```
 3 public class ContaPoupanca extends ContaBancaria{
 4
 5     public ContaPoupanca(String cliente, int numConta, float saldo){
 6         super(cliente, numConta, saldo);
 7     }
 8
 9     public void calcularNovoSaldo(float taxaRendimento){
10         this.setSaldo(this.getSaldo() + (this.getSaldo() * taxaRendimento));
11     }
12
13 }
```

Solução – Classe ContaEspecial

```
3  public class ContaEspecial extends ContaBancaria{
4
5      private float limite;
6
7      public ContaEspecial(String cliente, int numConta, float saldo, float limite){
8          super(cliente, numConta, saldo);
9          this.limite = limite;
10     }
11
12     public void sacar(float valor){
13         if(valor <= (this.getSaldo() + this.limite)){
14             this.setSaldo(this.getSaldo() - valor);
15         }
16         else{
17             System.out.println("Limite Ultrapassado - Saque Não Realizado");
18         }
19     }
20
21 }
```

Exercício 03 – Continuação

Após a implementação das classes acima, você deverá implementar uma classe Contas.Java, contendo o método main. Nesta classe, você deverá implementar:

a) Incluir dados relativos a(s) conta(s) de um cliente;
b) Sacar um determinado valor da(s) sua(s) conta(s);
c) Depositar um determinado valor na(s) sua(s) conta(s);
d) Mostrar o novo saldo do cliente, a partir da taxa de rendimento, daqueles que possuem conta poupança;
e) Mostrar os dados da(s) conta(s) de um cliente;

Exemplo - Classe Conta

```
3  public class Contas {
4
5      public static void main(String[] args) {
6
7          ContaPoupanca BBPoupanca = new ContaPoupanca("Fulano", 01, 1000);
8          ContaEspecial BBEspecial = new ContaEspecial("Beltrano", 02, 500, 100);
9
10         BBPoupanca.sacar(100);
11         BBEspecial.sacar(900);
12
13         System.out.println("Saldo após Saque");
14         System.out.println("Saldo BBPoupanca: " + BBPoupanca.getSaldo());
15         System.out.println("Saldo BBEspecial: " + BBEspecial.getSaldo());
16
17         BBPoupanca.depositar(100);
18         BBEspecial.depositar(100);
19
20         System.out.println("\nSaldo após Depósito");
21         System.out.println("Saldo BBPoupanca: " + BBPoupanca.getSaldo());
22         System.out.println("Saldo BBEspecial: " + BBEspecial.getSaldo());
23
24         System.out.println("\nSaldo após Rendimento");
25         BBPoupanca.calcularNovoSaldo((float) 0.5);
26         System.out.println("Saldo BBPoupanca: " + BBPoupanca.getSaldo());
27
28         //Método para mostrar os dados para os cliente
29         //Implementem
30     }
31 }
```

Extra - 04

Orientação a Objetos – DCC025

Prof. Edmar Welington Oliveira
oliveira.edmar@ufjf.edu.br

Universidade Federal de Juiz de Fora – UFJF
Departamento de Ciência da Computação - DCC

2011.3