



Tecnologia em Análise e Desenvolvimento de Sistemas

VITOR DA CUNHA DE SOUZA - RA: 3531963303

PORTFÓLIO - ROTEIRO DE AULA PRÁTICA LINGUAGEM ORIENTADA A OBJETOS

Pelotas - RS

2024

VITOR DA CUNHA DE SOUZA - RA: 3531963303

PORTFÓLIO - ROTEIRO DE AULA PRÁTICA LINGUAGEM ORIENTADA A OBJETOS

- Definição dos objetivos da aula prática:
Fixar os conceitos de programação orientada a objetos;
- Empregar práticas de instanciação de classes Java; Utilizar métodos e atributos em Java.
- Utilizar pacotes da linguagem Java para implementação de software com paradigma de orientação a objetos.

Orientadora: Marcio Akio Shimoda.

Pelotas - RS

2024

SUMÁRIO

| | | |
|------|---|----|
| 1. | PROBLEMA PROPOSTO | 4 |
| 2. | OBJETIVO GERAL | 5 |
| 3. | OBJETIVOS ESPECÍFICOS | 6 |
| 4. | PREPARANDO AMBIENTE | 7 |
| 4.1. | Instalação do Java SDK | 7 |
| 4.2. | Instalação do NetBeans | 7 |
| 5. | DESENVOLVENDO O CÓDIGO | 8 |
| 5.1. | Entendendo o código passo a passo | 8 |
| 5.2. | Vamos explicar o funcionamento do código .. | 8 |
| 5.3. | Ilustração do código | 10 |
| 6. | RESULTADO | 12 |
| 6.1. | Descrição do resultado | 12 |
| 6.2. | Ilustração do resultado | 14 |
| 7. | CONCLUSÃO | 16 |
| 8. | Referências | 17 |

1 PROBLEMA PROPOSTO

Este código é um exemplo simples de um sistema que ajuda a gerenciar um banco fictício através do terminal. Ele usa conceitos básicos de programação orientada a objetos, como criar classes e usar objetos para representar coisas do mundo real, como contas bancárias. É importante entender esses conceitos para fazer software robusto.

Para aprender mais sobre programação orientada a objetos em Java, recomendo ler livros como "Java: Como Programar" de Deitel & Deitel e "Use a Cabeça! Java" de Kathy Sierra e Bert Bates. Também há tutoriais online úteis, como os do W3Schools e da Oracle.

A programação orientada a objetos ajuda a reutilizar e manter o código, e este projeto mostra como isso funciona. Por exemplo, a classe `ContaBancaria` guarda informações de uma conta, o que facilita o reuso do código. Usar classes e objetos ajuda a tornar o sistema mais fácil de entender e manter.

Com base neste projeto, você pode construir sistemas mais complexos e avançados usando Java.

2 OBJETIVO GERAL

Atividade proposta: Usando os principais conceitos da Orientação a Objetos, faça um programa simples de gerenciamento bancário. Nele, o usuário pode inserir seu nome, sobrenome e CPF. A aplicação também deve permitir que o usuário veja o saldo, faça depósitos e saques. Essas ações devem continuar até que o usuário decida encerrar o programa.

3 OBJETIVOS ESPECÍFICOS

Utilizando os conceitos da Orientação a Objetos, crie um programa simples de gerenciamento bancário. Nele, o usuário pode inserir seu nome, sobrenome e CPF. O programa também deve permitir que o usuário veja o saldo, faça depósitos e saques. Essas ações devem continuar até que o usuário decida encerrar o programa.

Passos para criar o programa:

- 1 - No menu File, clique em New Project e escolha Java with Maven > Java Application.
- 2 - Dê um nome ao projeto, sugerindo "gerenciaBanco".
- 3 - Construa o programa em um único arquivo Java main Class, onde o método principal estará.
- 4 - No código, crie:
 - A. A classe principal
 - B. Uma classe para dados pessoais e operações bancárias
 - C. Um método para exibir o menu.
- 5 - Para mostrar o menu, use uma estrutura de decisão como do...while e switch..case para tratar as escolhas.

4 PREPARANDO AMBIENTE

4.1 Instalação do Java SDK

4.1.1 Acesse o site da Oracle e faça o download do Java SE Development Kit (JDK) mais recente.

4.1.2 Execute o instalador baixado e siga as instruções do assistente de instalação.

4.1.3 Após a instalação, é necessário configurar a variável de ambiente JAVA_HOME apontando para o diretório de instalação do JDK. Para isso, siga as instruções de acordo com o seu sistema operacional. Por exemplo, no Windows, você pode acessar as variáveis de ambiente nas configurações do sistema.

4.2 Instalação do NetBeans

Acesse o site do NetBeans e faça o download da versão mais recente do NetBeans IDE com suporte ao Java.

Execute o instalador baixado e siga as instruções do assistente de instalação. Durante a instalação, você pode ser solicitado a fornecer o caminho para a instalação do Java SDK. Certifique-se de apontar para o diretório onde o Java SDK foi instalado.

Após a conclusão da instalação, inicie o NetBeans e verifique se o Java SDK está corretamente configurado em "Ferramentas" > "Opções" > "Java" > "Plataformas". Se o Java SDK não estiver listado, adicione-o manualmente.

Estes passos devem permitir que você instale tanto o Java SDK quanto o NetBeans em seu sistema. Lembre-se de sempre baixar as versões mais recentes para garantir que você tenha acesso aos recursos e correções de segurança mais recentes.

5 DESENVOLVENDO O CÓDIGO

5.1 Entendendo o código passo a passo:

Este é um portfólio de aula prática que utiliza os conceitos da Orientação a Objetos para criar uma aplicação de gerenciamento bancário. O usuário pode inserir seu nome, sobrenome e CPF, além de consultar saldo, fazer depósitos e saques.

O código representa um sistema básico de gerenciamento bancário, permitindo que o usuário crie uma conta bancária fictícia e realize diversas operações como consultar saldo, fazer depósitos, saques e encerrar a sessão.

5.2 Vamos explicar o funcionamento do código:

Pacote e Importações:

O código começa importando a classe Scanner para permitir a entrada de dados pelo usuário.

Classe GerenciaBanco:

A classe principal que contém o método main.

No método main, é criada uma instância de Scanner para capturar a entrada do usuário e uma instância de ContaBancaria para representar a conta do usuário.

Entrada de Dados do Usuário:

O programa solicita ao usuário que digite seu nome, sobrenome e CPF.

Os dados são armazenados em variáveis locais.

Os métodos setarNome, setarSobrenome e setarCpf da classe ContaBancaria são chamados para definir esses valores na instância da conta.

Menu de Opções:

Após a configuração da conta, é exibido um menu em um loop do-while.

O usuário pode escolher entre as opções de consultar saldo, realizar depósito, realizar saque ou encerrar o programa.

Switch-Case para Opções:

O código utiliza um switch-case para lidar com cada opção escolhida pelo usuário.

Para cada opção, o programa executa a ação apropriada ou exibe uma mensagem de erro para opções inválidas.

Classe ContaBancaria (Classe Interna Estática):

Esta classe contém os detalhes da conta bancária, como nome, sobrenome, CPF e saldo.

Métodos como depositar e sacar permitem ao usuário adicionar e retirar dinheiro da conta, respectivamente.

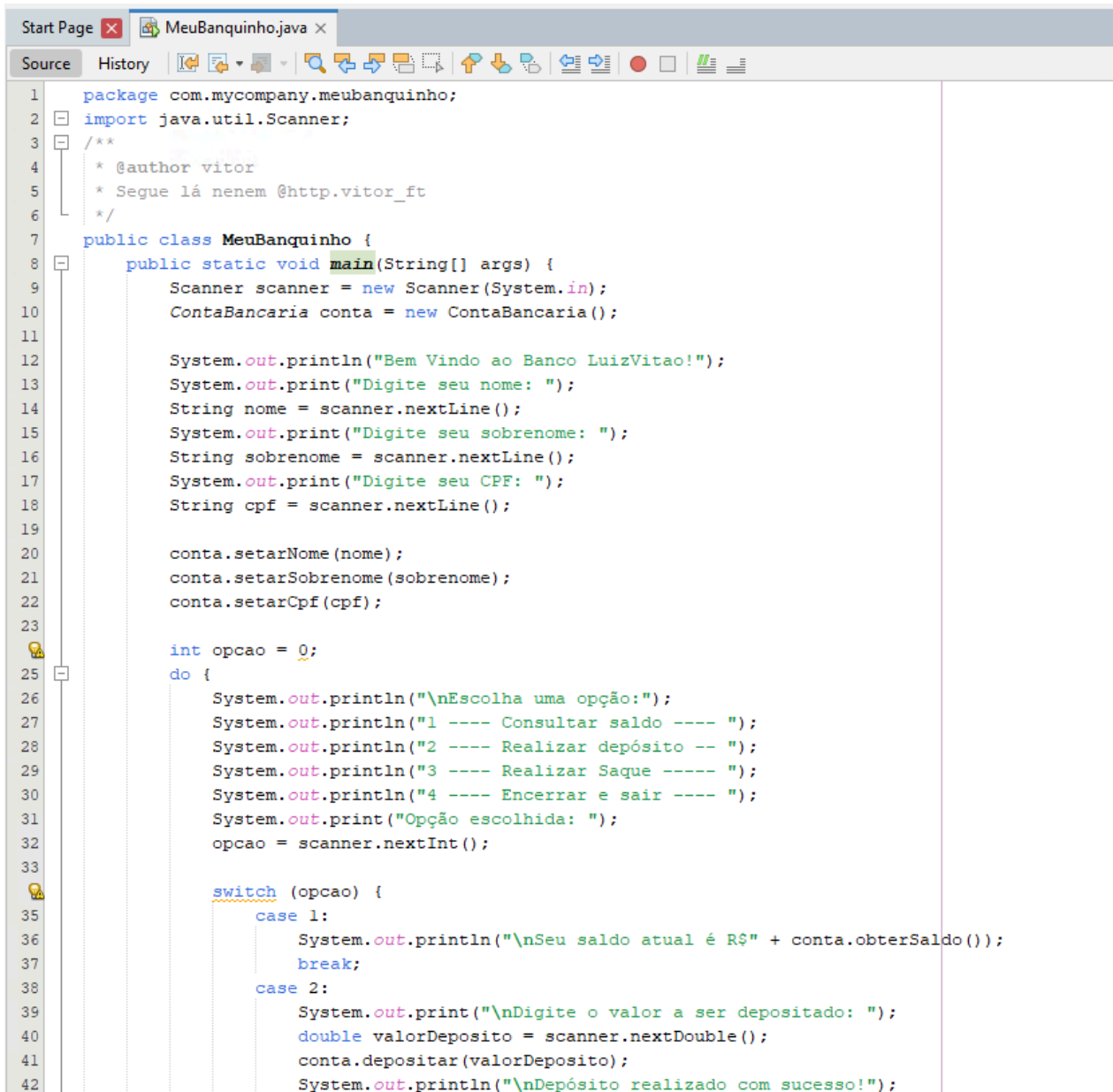
Os métodos setarNome, setarSobrenome e setarCpf são usados para configurar os detalhes da conta.

Finalização do Programa:

O programa continua a exibir o menu e aguardar as entradas do usuário até que a opção de encerrar seja escolhida.

Em resumo, este código implementa um simples sistema de gerenciamento de banco que permite ao usuário interagir com uma conta bancária fictícia, consultando saldo, depositando, sacando e encerrando a sessão quando desejar.

5.3. Ilustração do código:



```

1 package com.mycompany.meubanquinho;
2 import java.util.Scanner;
3 /**
4  * @author vitor
5  * Segue lá nenem @http.vitor_ft
6  */
7 public class MeuBanquinho {
8     public static void main(String[] args) {
9         Scanner scanner = new Scanner(System.in);
10        ContaBancaria conta = new ContaBancaria();
11
12        System.out.println("Bem Vindo ao Banco LuizVitao!");
13        System.out.print("Digite seu nome: ");
14        String nome = scanner.nextLine();
15        System.out.print("Digite seu sobrenome: ");
16        String sobrenome = scanner.nextLine();
17        System.out.print("Digite seu CPF: ");
18        String cpf = scanner.nextLine();
19
20        conta.setarNome(nome);
21        conta.setarSobrenome(sobrenome);
22        conta.setarCpf(cpf);
23
24        int opcao = 0;
25        do {
26            System.out.println("\nEscolha uma opção:");
27            System.out.println("1 ---- Consultar saldo ---- ");
28            System.out.println("2 ---- Realizar depósito -- ");
29            System.out.println("3 ---- Realizar Saque ----- ");
30            System.out.println("4 ---- Encerrar e sair ---- ");
31            System.out.print("Opção escolhida: ");
32            opcao = scanner.nextInt();
33
34            switch (opcao) {
35                case 1:
36                    System.out.println("\nSeu saldo atual é R$" + conta.obterSaldo());
37                    break;
38                case 2:
39                    System.out.print("\nDigite o valor a ser depositado: ");
40                    double valorDeposito = scanner.nextDouble();
41                    conta.depositar(valorDeposito);
42                    System.out.println("\nDepósito realizado com sucesso!");

```

```

Start Page x MeuBanquinho.java x
Source History
41      conta.depositar(valorDeposito);
42      System.out.println("\nDepósito realizado com sucesso!");
43      break;
44      case 3:
45          System.out.print("\nDigite o valor a ser sacado: ");
46          double valorSaque = scanner.nextDouble();
47          if (conta.sacar(valorSaque)) {
48              System.out.println("\nSaque realizado com sucesso!");
49          } else {
50              System.out.println("\nSaldo insuficiente!");
51          }
52          break;
53      case 4:
54          System.out.println("\nObrigado por utilizar o Banco LuizVitao!");
55          break;
56      default:
57          System.out.println("\nOpção inválida, por favor digite uma opção válida");
58      }
59      } while (opcao != 4);
60  }
61
62  static class ContaBancaria {
63      private String nome;
64      private String sobrenome;
65      private String cpf;
66      private double saldo;
67
68      public void setarNome(String nome) {
69          this.nome = nome;
70      }
71
72      public void setarSobrenome(String sobrenome) {
73          this.sobrenome = sobrenome;
74      }
75
76      public void setarCpf(String cpf) {
77          this.cpf = cpf;
78      }
79
80      public double obterSaldo() {
81          return saldo;
82      }
83
84      public void depositar(double valor) {
85          saldo += valor;
86      }
87
88      public boolean sacar(double valor) {
89          if (valor <= saldo) {
90              saldo -= valor;
91              return true;
92          } else {
93              return false;
94          }
95      }
96  }
97  }

```

6 RESULTADO

6.1 . Descrição do resultado

Ao executar este código no terminal, ocorre o seguinte processo:

O programa é compilado e executado.

Ao ser executado, o programa exibe a mensagem "Bem Vindo ao Banco LuizVítão!" e solicita ao usuário que forneça seu nome, sobrenome e CPF.

O usuário insere essas informações no terminal.

Depois que as informações são inseridas, o programa cria uma instância de ContaBancaria e atribui os valores de nome, sobrenome e CPF fornecidos pelo usuário a essa instância.

Em seguida, o programa exibe um menu com opções para o usuário escolher:

1. Consultar saldo
2. Realizar depósito
3. Realizar saque
4. Encerrar e sair

O usuário escolhe uma das opções digitando o número correspondente no terminal.

Dependendo da opção escolhida, o programa executa a operação correspondente:

Se o usuário escolher "Consultar saldo", o programa exibe o saldo atual da conta.

Se o usuário escolher "Realizar depósito", o programa solicita o valor a ser depositado e adiciona esse valor ao saldo da conta.

Se o usuário escolher "Realizar saque", o programa solicita o valor a ser sacado e verifica se o saldo é suficiente para realizar o saque. Se for, o saque é realizado e o saldo é atualizado. Caso contrário, o programa exibe uma mensagem informando que o saldo é insuficiente.

Se o usuário escolher "Encerrar e sair", o programa exibe uma mensagem de agradecimento e encerra a execução.

Após cada operação, o programa exibe novamente o menu de opções, permitindo que o usuário realize mais operações ou encerre o programa.

Esse ciclo continua até que o usuário escolha a opção de encerrar e sair (opção 4), momento em que o programa é finalizado.

Em resumo, o programa oferece uma interface simples para que o usuário gerencie uma conta bancária fictícia por meio do terminal.

6.2 Ilustração do resultado:

```

Start Page x MeuBanquinho.java x
Source History
29 System.out.println("3 ---- Realizar Saque ----- ");
com.mycompany.meubanquinho.MeuBanquinho > main > do ... while (opcao != 4) > switch (opcao) > case 4:
Output - Run (MeuBanquinho) x
Bem Vindo ao Banco LuizVital!
Digite seu nome: Vitor
Digite seu sobrenome: Souza
Digite seu CPF: 12312312312

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque -----
4 ---- Encerrar e sair ----
Opção escolhida: 2

Digite o valor a ser depositado: 250

Depósito realizado com sucesso!

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque -----
4 ---- Encerrar e sair ----
Opção escolhida: 1

Seu saldo atual é R$250.0

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque -----
4 ---- Encerrar e sair ----
Opção escolhida: 3

Digite o valor a ser sacado: 50

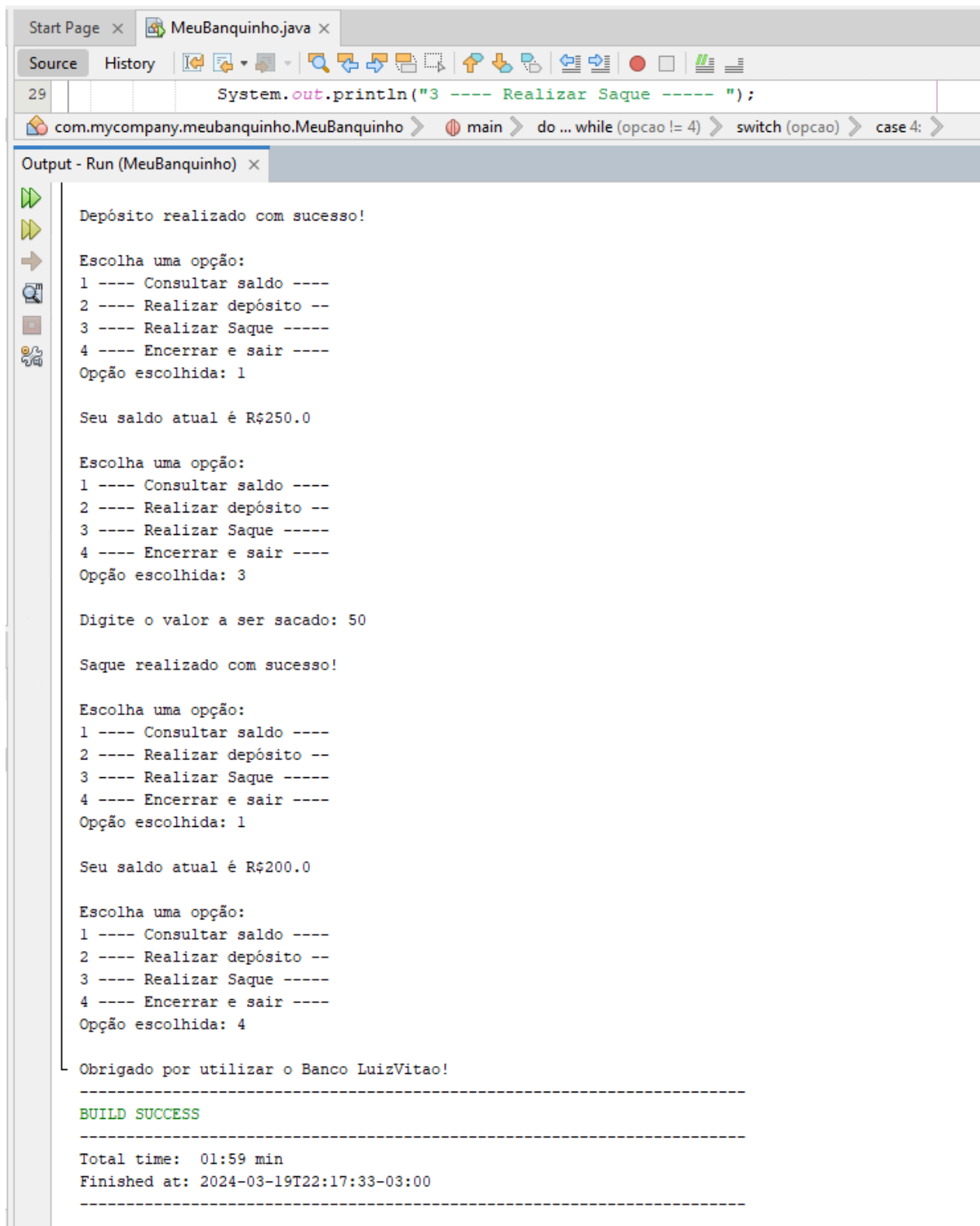
Saque realizado com sucesso!

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque -----
4 ---- Encerrar e sair ----
Opção escolhida: 1

Seu saldo atual é R$200.0

Escolha uma opção:
1 ---- Consultar saldo ----

```



```
Start Page x MeuBanquinho.java x
Source History
29 System.out.println("3 ---- Realizar Saque ---- ");
com.mycompany.meubanquinho.MeuBanquinho > main > do ... while (opcao != 4) > switch (opcao) > case 4: >
Output - Run (MeuBanquinho) x
Depósito realizado com sucesso!

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque ----
4 ---- Encerrar e sair ----
Opção escolhida: 1

Seu saldo atual é R$250.0

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque ----
4 ---- Encerrar e sair ----
Opção escolhida: 3

Digite o valor a ser sacado: 50

Saque realizado com sucesso!

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque ----
4 ---- Encerrar e sair ----
Opção escolhida: 1

Seu saldo atual é R$200.0

Escolha uma opção:
1 ---- Consultar saldo ----
2 ---- Realizar depósito --
3 ---- Realizar Saque ----
4 ---- Encerrar e sair ----
Opção escolhida: 4

Obrigado por utilizar o Banco LuizVitao!
-----
BUILD SUCCESS
-----
Total time: 01:59 min
Finished at: 2024-03-19T22:17:33-03:00
-----
```

7 CONCLUSÃO

O código fornecido representa uma aplicação básica de um sistema de gerenciamento bancário em Java. Essa aplicação permite ao usuário criar uma conta bancária fictícia e realizar diversas operações, como consultar saldo, fazer depósitos, efetuar saques e encerrar a sessão.

A implementação utiliza a classe Scanner para capturar a entrada de dados fornecida pelo usuário e a classe ContaBancaria para representar a conta do usuário. O menu de opções é exibido em um loop do-while e um switch-case é utilizado para lidar com cada escolha feita pelo usuário.

A classe ContaBancaria contém informações detalhadas da conta, como nome, sobrenome, CPF e saldo. Métodos como depositar e sacar permitem ao usuário adicionar e retirar dinheiro da conta, respectivamente.

Essa aplicação oferece uma interface simples para que o usuário possa gerenciar uma conta bancária fictícia por meio do terminal. Embora seja uma aplicação simples, o código exemplifica como os conceitos da programação orientada a objetos podem ser aplicados em um contexto real.

Além disso, é possível utilizar esse código como base para desenvolver uma aplicação mais complexa, com mais funcionalidades e recursos. Existem tutoriais e cursos online disponíveis que ensinam como desenvolver sistemas bancários em Java, o que pode ser uma excelente oportunidade para aprimorar as habilidades de programação e adquirir mais conhecimento sobre o paradigma de programação orientada a objetos.

Referências:**Java LTS Free.**

<https://www.oracle.com/br/java/technologies/downloads>

Apache NetBeans.

<https://netbeans.apache.org/front/main>