UNIVERSIDADE PITÁGORAS UNOPAR ANHANGUERA - MARAVILHA
ENGENHARIA DE SOFTWARE

NATAN OGLIARI - 34466876

# LINGUAGEM ORIENTADA A OBJETOS

Maravilha/SC
2023

NATAN OGLIARI - 34466876

# LINGUAGEM ORIENTADA A OBJETOS

Trabalho de portfólio apresentado como requisito parcial para a obtenção de pontos para a média semestral.

Orientador: Leonardo Santiago Sidon da Rocha.

Maravilha/SC
2023

# Sumário

# 1 Introdução

Esta presente aula prática tem por fim a aplicação dos paradigmas da linguagem orientada a objetos com a linguagem de programação Java1.
Para os procedimentos práticos foi sugerido o uso de nome *gerenciaBanco*, o qual será implementado. A finalidade desta aula prática, visa a implementação de um sistema de gerenciamento de banco, aonde o cliente deste banco irá acendero sistema atravéz de uma interface gráfica através da biblioteca *java.swing.*∗, o menu deverá ser do tipo *loop* o usuário deverá escolher a ação de finalizar a operação.


**Figura 1.** *Logo Java, Stackify (2023)*

As funções para este programa são:

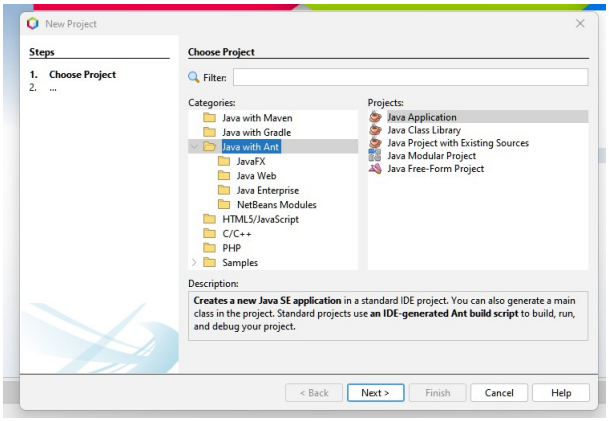
- Incerção das credenciais (Nome, Sobrenome e cpf);
- Consulta de saldo;
- Depósito;
- Saque;
- Finaliza a operação com uma mensagem de despedida;

Segundo freecodecamp (2023), a linguagem orientata a objetos se define em quatro pilares, sendo elas: herança, encapsulamento, abstração e poliformismo. Isso permite definir, neste caso, umas características de **cliente**, e utiliza-lós inúmeras vezes sem a necessidade de rescreve-lá.

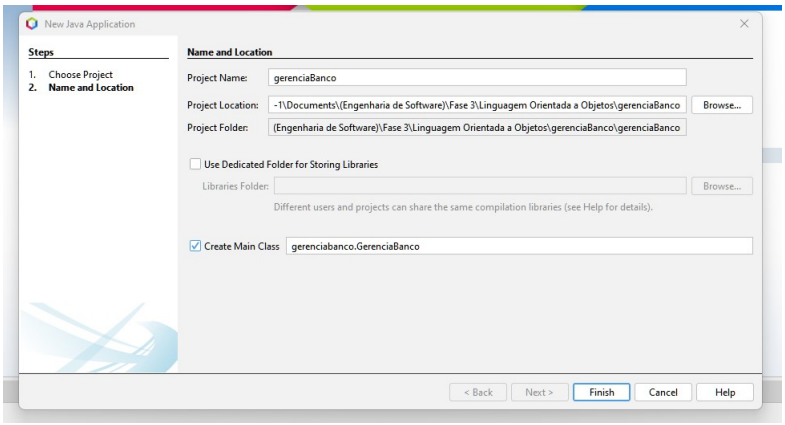
# 2 Métodos

Para a confecção desta aula prática, foi criado um repositório no **Github**, com o objetivo do aplicar os controle de versões e a utilização do sistema *git* em geral, o relatório foi confeccionado em LaTeX com o objetivo da abstração da formatação do texto ficando livre para o aprofundamento na questão textual.

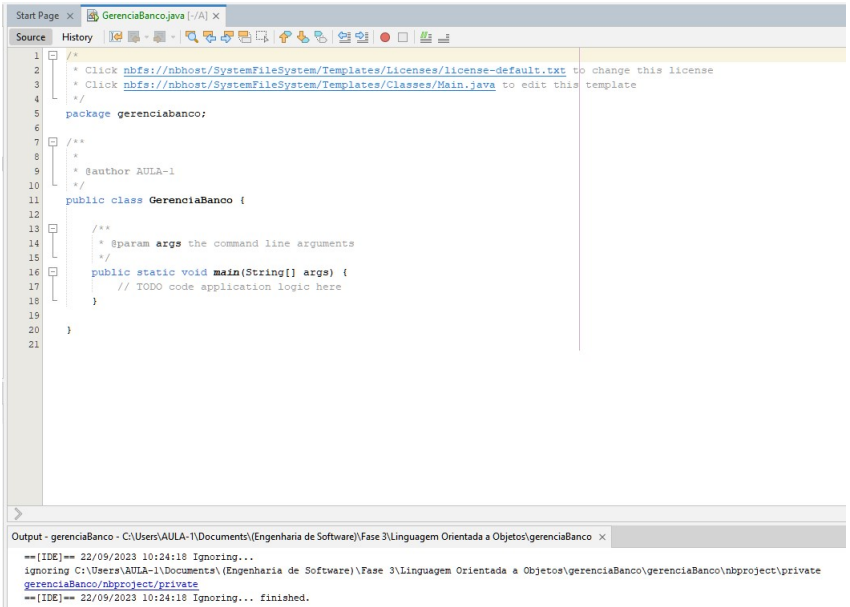
De forma subsequente as instruções do roteiro da aula prática é criado um projeto no NetBeans em java conforme figura 2.




(a) Início projeto.




(b) Nome projeto.




(c) Projeto em branco.

**Figura 2.** *Projeto java, O autor (2023)*

Na figura 2(a) solicita o tipo de projeto a ser criado, no caso *Java Application*, na figura 2(b) é solicitado o local aonde o projeto será salvo e o nome do mesmo, deste modo o nome foi

definido conforme sugestão do roteiro *gerenciaBanco* e na figura 2(c) demonstra a criação do projeto em **branco** já com a inclusão do pacote *gerenciabanco*,após estes passos foi prossegido com a programação do que foi solicitado ao discente.

Para organização desta aula prática foi confeccionado um diagrama UML para o programa, conforme demonstra a figura 3

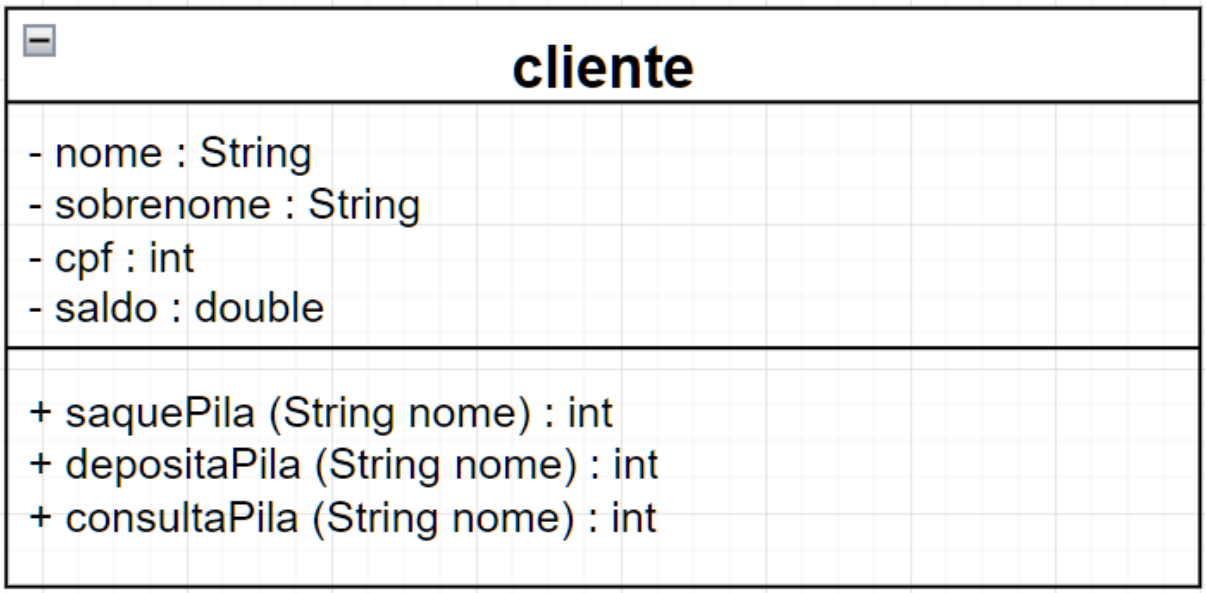



**Figura 3.** *Diagrama UML, O autor (2023)*

Forãm definidos atributos para a classe e o mesmos são:

- name : String
- sobrenome : String
- cpf : String
- saldo : double

Seguindo boas práticas de programação, define-se variáveis de controle de erros, e para tal, a criação de constatnte de verificação em vez de zeros e uns (0,1), do tipo $privatestaticfinalintSTATUS\_OK = 1$; e $privatestaticfinalintSTATUS\_FAIL = 0$;.

# 3 Resultados

Após a criação dos repositórios e do projeto, prossegue com a confecção do código[1], no decorrer desta seção será incluido figuras dos resultados da presente aula.

## 3.1 Métodos das classes

Os métodos definem as ações da classe, e o acesso é $minhaClasse.meuMetodo(args)$

```
1     /**
2     * Realiza a consulta do saldo de uma conta
3     * @param nome Passa o nome da conta a ser consultada e informa o
          saldo ao cliente.
4     */
5      public double consultaPilas(String nome){
6      JOptionPane.showMessageDialog(null,"Seu saldo atual e de: R$" +
          this.saldo, this.nome+" "+this.sobrenome, JOptionPane.
          INFORMATION_MESSAGE, icon);
7      }
```

**Listing 1.** *consultaPilas*

```
1   /**
2    * Realiza um depósito em uma conta
3    * @param nome Informa o nome da conta a ser depositado
4    * @return retorna STATUS_OK se a operacao ocorreu com sucesso e
        retorna STATUS_FAIL se ocorrer um erro
5    */
6   public void depositaPila( String nome){
```

[1] código completo

```
7           try {// verifica se a entrada e do tipo numeral
8               double pilaDeposito = Double.parseDouble(JOptionPane.
                   showInputDialog(null,"Informe a quantidade em Reais
                   (R$) a ser depositado na conta do "+this.nome+" "+
                   this.sobrenome));
9                this.saldo += pilaDeposito;
10              //JOptionPane.showMessageDialog(null,"Seu Saldo e R$
                   :"+this.consultaPilas(this.nome)+" Reais", this.
                   nome, JOptionPane.INFORMATION_MESSAGE, icon);
11              return STATUS_OK;
12           }
13
14           catch (NumberFormatException e) {// imprime o erro na tela
                e informa o que foi digitado.
15              JOptionPane.showMessageDialog(null,"Entre com valor
                   valido, do tipo numeral.\n Use (.) ponto em vez de
                   (,) virgula\n ERRO: " + e.getMessage()  , "ERRO",
                   JOptionPane.ERROR_MESSAGE);
16              return STATUS_FAIL;
17           }
18   }
```

**Listing 2.** *depositoPilas*

# 4 Conclusões

Esta aula prática teve como fim a aplicação de conhecimentos adquiridos em sala de aula virtuais da disciplina de linguagem orientada a objetos do curso bacharel em engenharia de software da faculdade UNOPAR,

o código completo[2] pode ser visto nos anexos5 deste documento.

[2]código completo

# Referências

FREECODECAMP. **Programação orientada a objetos e programação estruturada**. 2023. Acessado em: 06 nov. 2023. Disponível em: <https://www.freecodecamp.org/portuguese/news/os-quatro-pilares-da-programacao-orientada-a-objetos-com-javascript/>.

STACKIFY. **Application Logging e Exception Handling**. 2023. Acessado em: 09 set. 2023. Disponível em: <https://stackify.com/java-logging-best-practices/>.

# 5 Anexos

```
1  package gerenciabanco;
2
3  import java.util.*;
4  //import javax.swing.JOptionPane;
5  import javax.swing.*;
6  import java.awt.*;
7  import java.lang.Integer;
8  import java.lang.Exception;
9  import java.lang.Error;
10 import java.lang.reflect.Method;
11 /**
12  * @author Natan Ogliari
13  * @version 0.1
14  */
15 public class GerenciaBanco {
16
17     public static class cliente{
18         String nome;
19         String sobrenome;
20         String cpf;
21         double saldo;
22
23         private static final int STATUS_OK = 1;//!<define um retorno
               caso de sucesso>
24         private static final int STATUS_FAIL = 0;//!<define um
               retorno caso de falha>
25
26         ImageIcon icon = new ImageIcon("C:\\Users\\AULA-1\\Documents
               \\(Engenharia de Software)\\Fase 3\\Linguagem Orientada a
               Objetos\\gerenciaBanco\\gerenciaBanco\\src\\gerenciabanco
               \\saracura.jpg");
```

```
27
28          /**
29          * Realiza um deposito em uma conta
30          * @param nome Informa o nome da conta a ser depositado
31          * @return retorna STATUS_OK se a operação ocorreu com sucesso
                e retorna STATUS_FAIL se ocorrer um erro
32          */
33          public int depositaPila(String nome){
34
35              try {// verifica se a entrada e do tipo numeral
36                 double pilaDeposito = Double.parseDouble(JOptionPane.
                       showInputDialog(null,"Informe a quantidade em Reais
                        (R$) a ser depositado na conta do "+this.nome+" "+
                       this.sobrenome));
37                  this.saldo += pilaDeposito;
38                 //JOptionPane.showMessageDialog(null,"Seu Saldo é R$
                       :"+this.consultaPilas(this.nome)+" Reais", this.
                       nome, JOptionPane.INFORMATION_MESSAGE, icon);
39                  return STATUS_OK;
40              }
41
42              catch (NumberFormatException e) {// imprime o erro na
                   tela e informa o que foi digitado.
43                  JOptionPane.showMessageDialog(null,"Entre com valor
                       válido, do tipo númeral.\n Use (.) ponto em vez de
                        (,) virgula\n ERRO: " + e.getMessage()   , "ERRO",
                        JOptionPane.ERROR_MESSAGE);
44                  return STATUS_FAIL;
45              }
46          }
47
48          /**
49          * Realiza o saque de uma conta
50          * @param nome Informa o nome da conta a ser realizado o saque
51          * @return retorna STATUS_OK se a operação ocorreu com sucesso
                e retorna STATUS_FAIL se ocorrer um erro
52          */
53          public int saquePila(String nome){
54
55              try {// tratamento de exception
56                      double pilaSaque = Double.parseDouble(JOptionPane
```

```
                        .showInputDialog(null,"Informe a quantidade em
                         Reais (R$) a ser sacada na conta do "+this.
                        nome+" "+this.sobrenome));
57                  if (this.saldo >= pilaSaque){//verifica se tem
                        saldo
58                      this.saldo -= pilaSaque;
59                      return STATUS_OK;
60                  }
61                  else{
62                      JOptionPane.showMessageDialog(null,"O valor
                            informado para o saque é maior que seu
                            saldo.\nSeu saldo é de: R$ "+this.saldo, "
                            Saldo Insuficiente", JOptionPane.
                            ERROR_MESSAGE, icon);
63                      return STATUS_FAIL;
64                  }
65              }
66              catch (NumberFormatException e){
67                      JOptionPane.showMessageDialog(null,"Entre com
                            valor válido, do tipo númeral.\n ERRO: " + e.
                            getMessage()  , "ERRO", JOptionPane.
                            ERROR_MESSAGE);
68              return STATUS_FAIL;
69              }
70          }
71
72          /**
73          * Realiza a consulta do saldo de uma conta
74          * @param nome Passa o nome da conta a ser consultada e
                informa o saldo ao cliente.
75          */
76          public void consultaPilas(String nome){
77              JOptionPane.showMessageDialog(null,"Seu saldo atual é de:
                     R$" +this.saldo, this.nome+" "+this.sobrenome,
                    JOptionPane.INFORMATION_MESSAGE, icon);
78          }
79      }
80
81      public static void main(String[] args) {
82
83          ImageIcon icon = new ImageIcon("C:\\Users\\AULA-1\\Documents
```

```
            \\(Engenharia de Software)\\Fase 3\\Linguagem Orientada a
            Objetos\\gerenciaBanco\\gerenciaBanco\\src\\gerenciabanco
            \\saracura.jpg");
84
85          cliente cliente1 = new cliente();//instância o cliente
86          JOptionPane.showMessageDialog(null,"Bem vindo ao Banco
               Saracura do Banhado\n", "INÍCIO", JOptionPane.
               INFORMATION_MESSAGE, icon);//add custon icon
87          try {
88              cliente1.nome = JOptionPane.showInputDialog(null, "
                   Informe seu Nome.", "Nome");
89              if (cliente1.nome  == null){//caso o usuario cancele a
                   opção
90                  JOptionPane.showMessageDialog(null, "Você cancelou a
                       operação");
91              }
92          }
93          catch (NullPointerException e){
94              JOptionPane.showMessageDialog(null, "Entre com um Nome
                   válido.","Erro" , JOptionPane.ERROR_MESSAGE);
95          }
96
97          try {
98              cliente1.sobrenome = JOptionPane.showInputDialog(null, "
                   Informe seu Sobrenome.","Sobrenome");
99          }
100         catch(NullPointerException e){
101             JOptionPane.showMessageDialog(null, "Entre com um
                   sobrenome válido.","Erro" , JOptionPane.ERROR_MESSAGE)
                   ;
102         }
103
104         try{
105             cliente1.cpf = JOptionPane.showInputDialog(null, "Informe
                    o numéro do CPF.","000.000.000-00");
106         }
107         catch (NullPointerException e){
108             JOptionPane.showMessageDialog(null, "Entre com um CPF
                   válido.","Erro" , JOptionPane.ERROR_MESSAGE);
109         }
110         //cliente1.saldo = 445;//remover
```

```
111         while(true){ //InterruptedIOException
112
113             String opcao = JOptionPane.showInputDialog(null,"Opção 1
                    - Consulta saldo\n Opção 2 - Realizar um deposito\n
                    Opção 3 - Realizar um saque\n Opção 4 - Sair \n",4);//
                    deixa a opçã4 4 como defu l
114             //conversão de String para int
115             int control = 0;
116             if ("opcao" == null){
117
118                 control = 0;
119             }
120             else {
121                  control = Integer.parseInt(opcao);
122             }
123
124             //int control = Integer.parseInt(opcao);
125
126             if ("opcao"  == null){//caso o usuario cancele a opção
127                 JOptionPane.showMessageDialog(null, "Você cancelou a
                        operação");
128                 break;
129             }
130
131             if (control == 0){
132                 JOptionPane.showMessageDialog(null, "Você cancelou a
                        operação");
133                 break;
134             }
135
136             if (control == 4){//para sair da operação
137                 JOptionPane.showMessageDialog(null,"Volte sempre "+
                        cliente1.nome+" "+cliente1.sobrenome+"\nTenha um
                        bom dia!", "LOGOUT", JOptionPane.
                        INFORMATION_MESSAGE, icon);//add custon icon
138                 break;
139             }
140
141             switch (control){
142                 case 0:
143                     JOptionPane.showMessageDialog(null, "Você
```

```
                    cancelou a operação");
144                 break;
145
146             case 1://consulta de saldo   consultaPilas(nome)
147                 JOptionPane.showMessageDialog(null,"Consulta
                        saldo.", cliente1.nome+" "+cliente1.sobrenome,
                         JOptionPane.INFORMATION_MESSAGE, icon);//add
                        custon icon
148                 cliente1.consultaPilas(cliente1.nome);
149                 break;
150
151             case 2://realiza deposito
152                 JOptionPane.showMessageDialog(null,"Realizar um
                        deposito.", cliente1.nome+" "+cliente1.
                        sobrenome, JOptionPane.INFORMATION_MESSAGE,
                        icon);
153                 int verifica = cliente1.depositaPila(cliente1.
                        nome);
154                 if (verifica == 1){
155                     JOptionPane.showMessageDialog(null,"Depósito
                            realizado com sucesso.", cliente1.nome+" "
                            +cliente1.sobrenome, JOptionPane.
                            INFORMATION_MESSAGE, icon);
156                 }else{
157                     JOptionPane.showMessageDialog(null,"Falha no
                            deposito.\n Tente novamente.", cliente1.
                            nome+" "+cliente1.sobrenome, JOptionPane.
                            ERROR_MESSAGE, icon);
158                 }
159                 break;
160
161             case 3:// realiza saque
162                 JOptionPane.showMessageDialog(null,"Realiza um
                        saque.", cliente1.nome+" "+cliente1.sobrenome,
                         JOptionPane.INFORMATION_MESSAGE, icon);
163                 verifica = cliente1.saquePila(cliente1.nome);
164
165                 if (verifica == 1){
166                     JOptionPane.showMessageDialog(null,"Saque
                            realizado com sucesso.", cliente1.nome+" "
                            +cliente1.sobrenome, JOptionPane.
```

```
                                INFORMATION_MESSAGE, icon);
167                         }else{
168                             JOptionPane.showMessageDialog(null,"Falha no
                                saque.\n Tente novamente.", cliente1.nome+
                                " "+cliente1.sobrenome, JOptionPane.
                                ERROR_MESSAGE, icon);
169                         }
170                         break;
171
172                     default:
173                         JOptionPane.showMessageDialog(null,"Opção
                            inválida.", cliente1.nome+" "+cliente1.
                            sobrenome, JOptionPane.INFORMATION_MESSAGE,
                            icon);
174                         break;
175                 }
176             }
177         }
178 }
```