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:

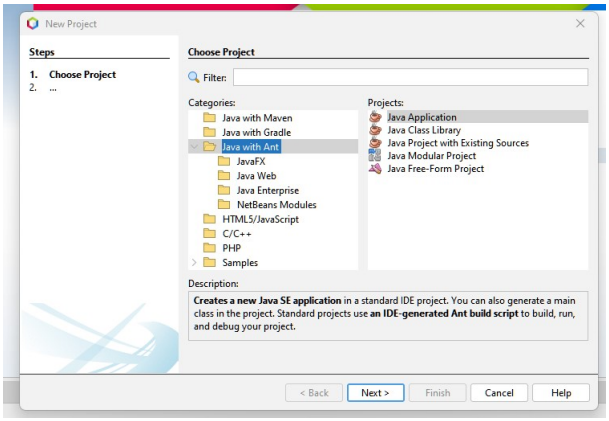
- Incerção das credenciais (Nome, Sobrenome e cpf);
- Consulta de saldo;
- Depósito;
- Saque;
- Finaliza a operação com uma mensagem de despedida;

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.

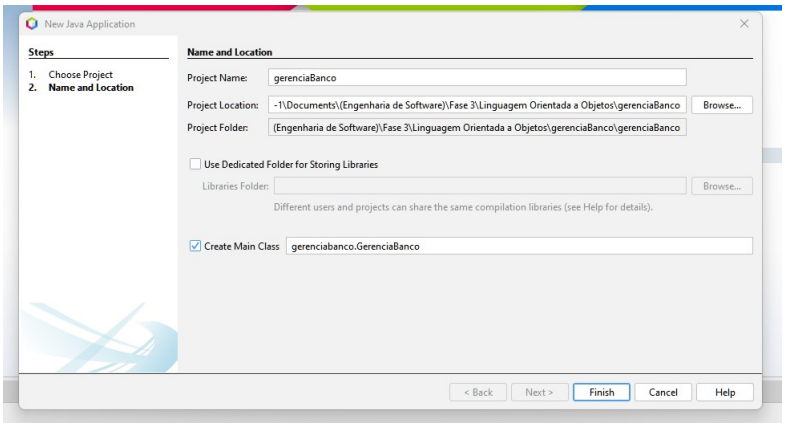
2 Métodos

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.

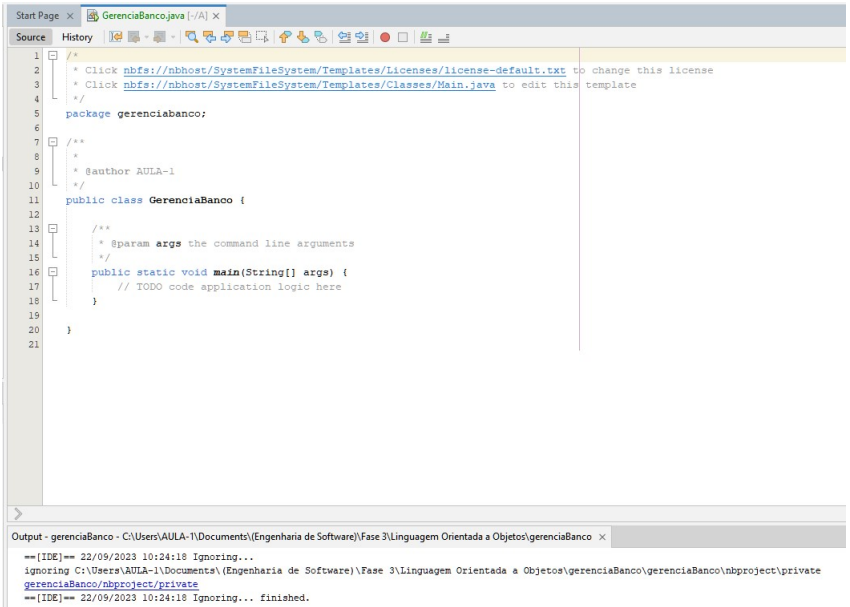
De forma subsequente as instruções do roteiro da aula prática é criado um projeto no NetBeans em java conforme figura 2.



(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

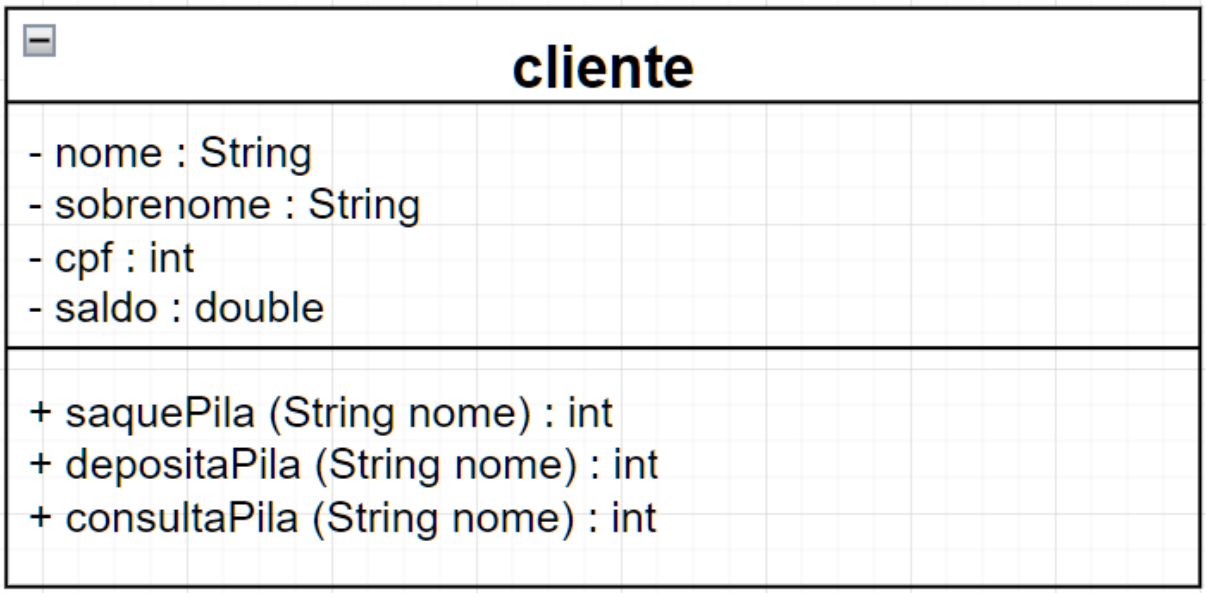


Figura 3. *Diagrama UML, O autor (2023)*

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 $private static final int STATUS\_OK = 1$; e $private static final int STATUS\_FAIL = 0$;.

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 $minhaClasse.meuMetodo(args)$, foram definidos três métodos, sendo eles:

- *consultaPila()* Responsavél pela conulta do saldo do cliente, conforme código 1;
- *depositoPila()* Responsavél pelo depósito na conta do cliente, conforme código 2 e
- *saquePila()* Responsavél pelo saque na conta do clinte, conforme código 3.

```
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 void consultaPila(String nome){
6             JOptionPane.showMessageDialog(null,"Seu saldo atual é de:
                  R$" +this.saldo, this.nome+" "+this.sobrenome,
                 JOptionPane.INFORMATION_MESSAGE, icon);
7         }
8     }
```

Listing 1. *consultaPila()*

[1] código completo

```
1         /**
2         * Realiza um deposito em uma conta
3         * @param nome Informa o nome da conta a ser depositado
4         * @return retorna STATUS_OK se a operação ocorreu com sucesso
              e retorna STATUS_FAIL se ocorrer um erro
5         */
6         public int depositaPila(String nome){
7
8             try {// verifica se a entrada e do tipo numeral
9                double pilaDeposito = Double.parseDouble(JOptionPane.
                    showInputDialog(null,"Informe a quantidade em Reais
                     (R$) a ser depositado na conta do "+this.nome+" "+
                    this.sobrenome));
10                this.saldo += pilaDeposito;
11               //JOptionPane.showMessageDialog(null,"Seu Saldo é R$
                    :"+this.consultaPilas(this.nome)+" Reais", this.
                    nome, JOptionPane.INFORMATION_MESSAGE, icon);
12                return STATUS_OK;
13            }
14
15            catch (NumberFormatException e) {// imprime o erro na
                 tela e informa o que foi digitado.
16                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);
17                return STATUS_FAIL;
18            }
19        }
```

Listing 2. *depositoPila()*

```
1         /**
2         * Realiza o saque de uma conta
3         * @param nome Informa o nome da conta a ser realizado o saque
4         * @return retorna STATUS_OK se a operação ocorreu com sucesso
              e retorna STATUS_FAIL se ocorrer um erro
5         */
6         public int saquePila(String nome){
7
8             try {// tratamento de exception
9                    if (this.saldo > 0){
10                       return STATUS_FAIL;
```

```
11                  }
12                   double pilaSaque = Double.parseDouble(JOptionPane
                        .showInputDialog(null,"Informe a quantidade em
                         Reais (R$) a ser sacada na conta do "+this.
                        nome+" "+this.sobrenome));
13                   if (this.saldo >= pilaSaque){//verifica se tem
                        saldo
14                       this.saldo -= pilaSaque;
15                       return STATUS_OK;
16                   }
17                   else{
18                       JOptionPane.showMessageDialog(null,"O valor
                            informado para o saque é maior que seu
                            saldo.\nSeu saldo é de: R$ "+this.saldo, "
                            Saldo Insuficiente", JOptionPane.
                            ERROR_MESSAGE, icon);
19                       return STATUS_FAIL;
20                   }
21           }
22           catch (NumberFormatException e){
23                   JOptionPane.showMessageDialog(null,"Entre com
                        valor válido, do tipo númeral.\n ERRO: " + e.
                        getMessage()  , "ERRO", JOptionPane.
                        ERROR_MESSAGE);
24           return STATUS_FAIL;
25           }
26       }
```

Listing 3. *saquePila()*

É solicitado que crie um *loop*, para que o usuário escolha quando deseja sair da operação, portanto o código para tal é o código 4 e a figura 4.

```
1        while(true){ //InterruptedIOException
2
3            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 deful
4            //conversão de String para int
5            int control = 0;
6            if ("opcao" == null){
7
8                control = 0;
```

```
9             }
10            else {
11                 control = Integer.parseInt(opcao);
12            }
13
14            //int control = Integer.parseInt(opcao);
15
16            if ("opcao"  == null){//caso o usuario cancele a opção
17                JOptionPane.showMessageDialog(null, "Você cancelou a
                      operação");
18                break;
19            }
20
21            if (control == 0){
22                JOptionPane.showMessageDialog(null, "Você cancelou a
                      operação");
23                break;
24            }
25
26            if (control == 4){//para sair da operação
27                JOptionPane.showMessageDialog(null,"Volte sempre "+
                      cliente1.nome+" "+cliente1.sobrenome+"\nTenha um
                      bom dia!", "LOGOUT", JOptionPane.
                      INFORMATION_MESSAGE, icon);//add custon icon
28                break;
29            }
30
31            switch (control){
32                case 0:
33                    JOptionPane.showMessageDialog(null, "Você
                          cancelou a operação");
34                    break;
35
36                case 1://consulta de saldo   consultaPilas(nome)
37                    JOptionPane.showMessageDialog(null,"Consulta
                          saldo.", cliente1.nome+" "+cliente1.sobrenome,
                           JOptionPane.INFORMATION_MESSAGE, icon);//add
                          custon icon
38                    cliente1.consultaPila(cliente1.nome);
39                    break;
40
```

```
41                  case 2://realiza deposito
42                      JOptionPane.showMessageDialog(null,"Realizar um
                            deposito.", cliente1.nome+" "+cliente1.
                            sobrenome, JOptionPane.INFORMATION_MESSAGE,
                            icon);
43                      int verifica = cliente1.depositaPila(cliente1.
                            nome);
44                      if (verifica == 1){
45                          JOptionPane.showMessageDialog(null,"Depósito
                                realizado com sucesso.", cliente1.nome+" "
                                +cliente1.sobrenome, JOptionPane.
                                INFORMATION_MESSAGE, icon);
46                      }else{
47                          JOptionPane.showMessageDialog(null,"Falha no
                                deposito.\n Tente novamente.", cliente1.
                                nome+" "+cliente1.sobrenome, JOptionPane.
                                ERROR_MESSAGE, icon);
48                      }
49                      break;
50
51                  case 3:// realiza saque
52                      JOptionPane.showMessageDialog(null,"Realiza um
                            saque.", cliente1.nome+" "+cliente1.sobrenome,
                            JOptionPane.INFORMATION_MESSAGE, icon);
53                      verifica = cliente1.saquePila(cliente1.nome);
54
55                      if (verifica == 1){
56                          JOptionPane.showMessageDialog(null,"Saque
                                realizado com sucesso.", cliente1.nome+" "
                                +cliente1.sobrenome, JOptionPane.
                                INFORMATION_MESSAGE, icon);
57                      }else{
58                          JOptionPane.showMessageDialog(null,"Falha no
                                saque.\n Tente novamente.", cliente1.nome+
                                " "+cliente1.sobrenome, JOptionPane.
                                ERROR_MESSAGE, icon);
59                      }
60                      break;
61
62                  default:
63                      JOptionPane.showMessageDialog(null,"Opção
```

```
                        inválida.", cliente1.nome+" "+cliente1.
                        sobrenome, JOptionPane.INFORMATION_MESSAGE,
                        icon);
64                  break;
65              }
66          }
```

Listing 4. *loop*

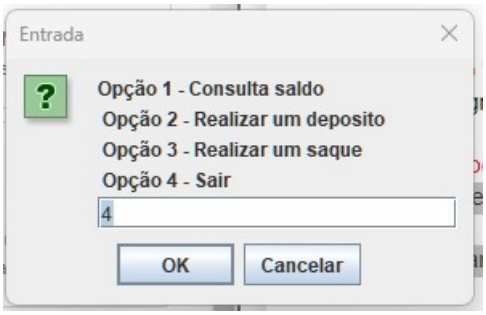
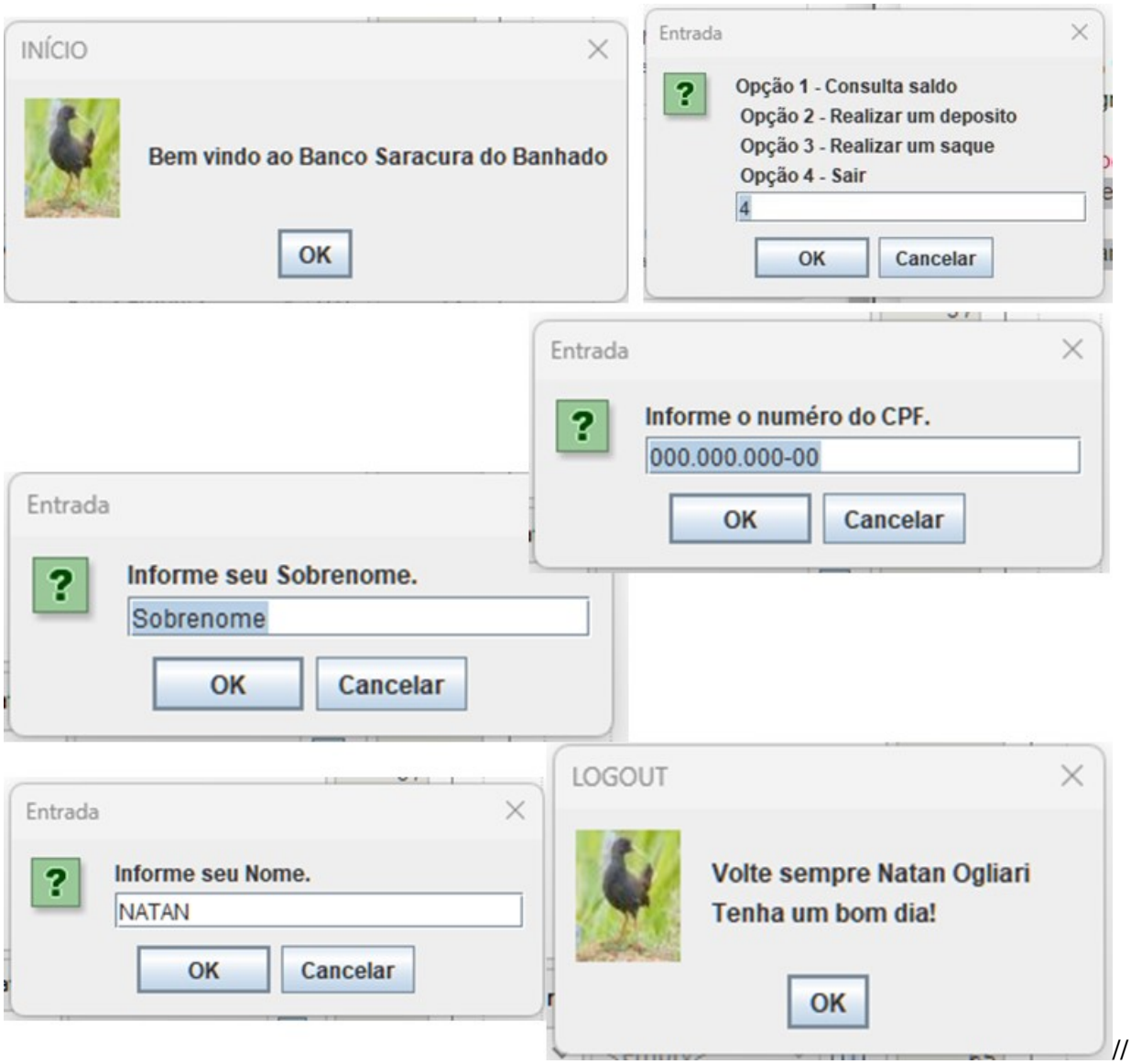
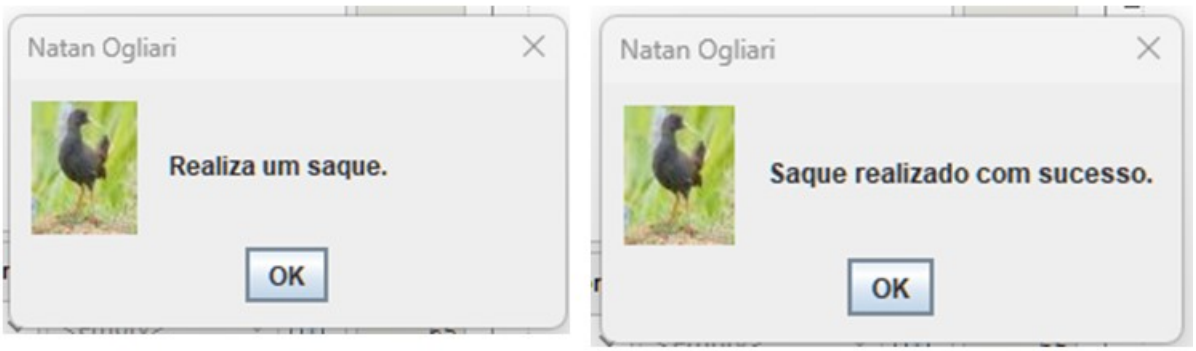


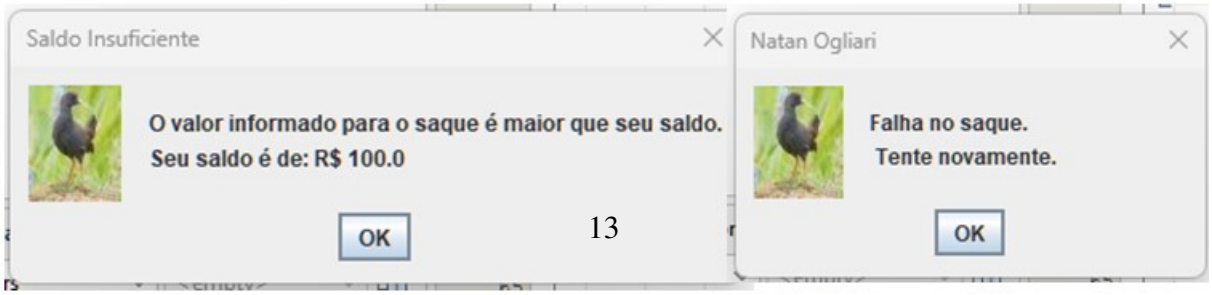
Figura 4. *Interface gráfica do loop, O autor(2023)*

Foi confeccionado uma interface gráfica para o usuário, conforme demonstra as figuras 5

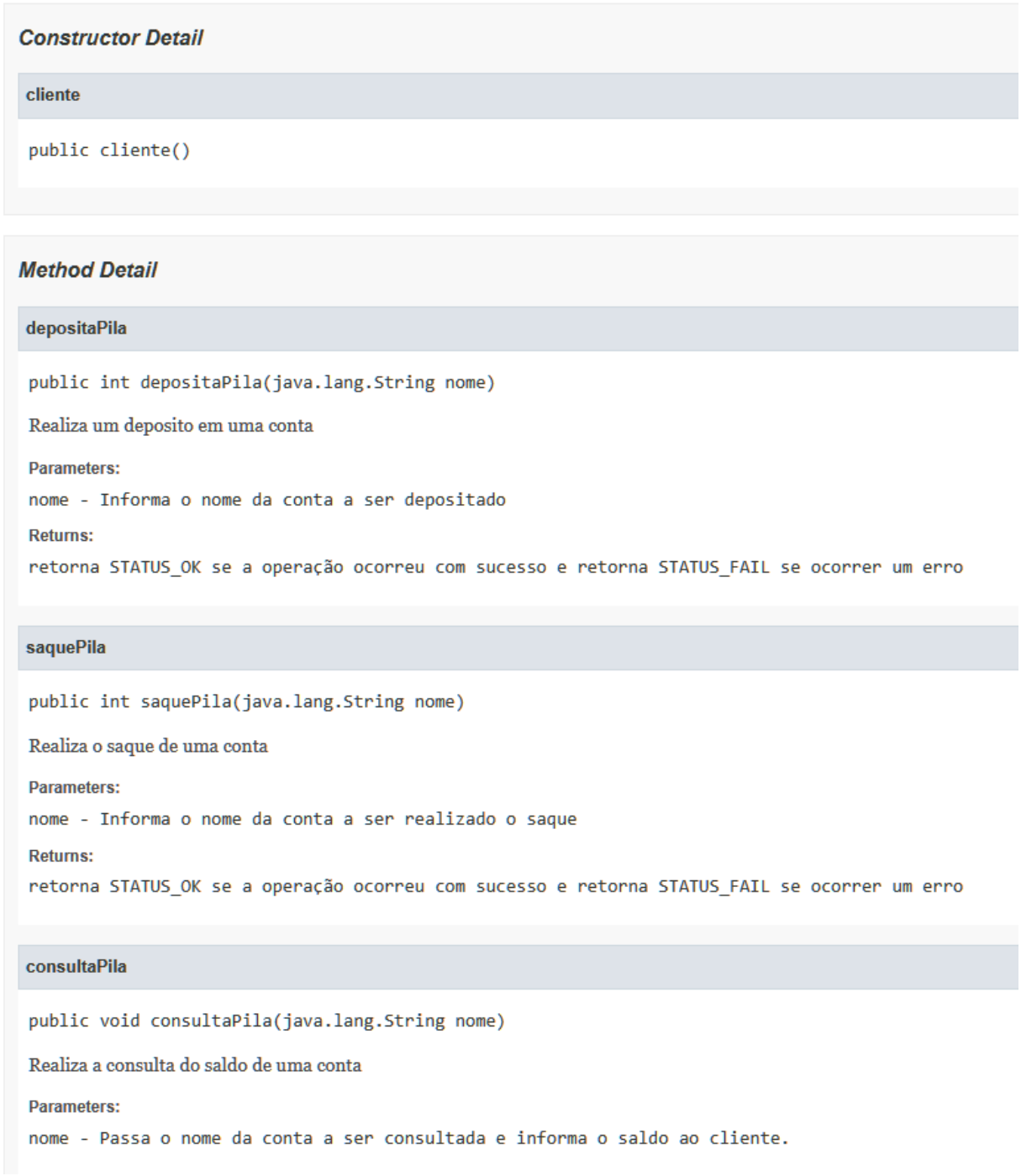

INÍCIO
Bem vindo ao Banco Saracura do Banhado
OK
Entrada
Opção 1 - Consulta saldo
Opção 2 - Realizar um deposito
Opção 3 - Realizar um saque
Opção 4 - Sair
4
OK
Cancelar
Entrada
Informe o numéro do CPF.
000.000.000-00
OK
Cancelar
Entrada
Informe seu Sobrenome.
Sobrenome
OK
Cancelar
Entrada
Informe seu Nome.
NATAN
OK
Cancelar
LOGOUT
Volte sempre Natan Ogliari
Tenha um bom dia!
OK

(a) Menu total.


Natan Ogliari
Realiza um saque.
OK
Natan Ogliari
Saque realizado com sucesso.
OK


Saldo Insuficiente
O valor informado para o saque é maior que seu saldo.
Seu saldo é de: R$ 100.0
OK
Natan Ogliari
Falha no saque.
Tente novamente.
OK

Para este projeto foi utilizado os comentários no formato do *javadoc*, e conforme a figura **??**, demonstra a documentação deste projeto gerada automaticamente pelo NetBeans.



Constructor Detail

cliente

```
public cliente()
```

Method Detail

depositaPila

```
public int depositaPila(java.lang.String nome)
```

Realiza um deposito em uma conta

Parameters:

`nome - Informa o nome da conta a ser depositado`

Returns:

`retorna STATUS_OK se a operação ocorreu com sucesso e retorna STATUS_FAIL se ocorrer um erro`

saquePila

```
public int saquePila(java.lang.String nome)
```

Realiza o saque de uma conta

Parameters:

`nome - Informa o nome da conta a ser realizado o saque`

Returns:

`retorna STATUS_OK se a operação ocorreu com sucesso e retorna STATUS_FAIL se ocorrer um erro`

consultaPila

```
public void consultaPila(java.lang.String nome)
```

Realiza a consulta do saldo de uma conta

Parameters:

`nome - Passa o nome da conta a ser consultada e informa o saldo ao cliente.`

Figura 6. *Documentação do projeto pelo javadoc, O autor (2023)*

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 anexos 5 deste documento.

Com a implementação desta aula prática ficou evidente que a é importante o tratamento de erros, principalmente

[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                     if (this.saldo > 0){
```

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

```
84      public static void main(String[] args) {
85          /**Importa o logo do banco. */
86          ImageIcon icon = new ImageIcon("C:\\Users\\AULA-1\\Documents
                \\(Engenharia de Software)\\Fase 3\\Linguagem Orientada a
                Objetos\\gerenciaBanco\\gerenciaBanco\\src\\gerenciabanco
                \\saracura.jpg");
87
88          /** Instância a classe */
89          cliente cliente1 = new cliente();
90          JOptionPane.showMessageDialog(null,"Bem vindo ao Banco
                Saracura do Banhado\n", "INÍCIO", JOptionPane.
                INFORMATION_MESSAGE, icon);//add custon icon
91          try {/** customer's name and checks for possible errors */
92              cliente1.nome = JOptionPane.showInputDialog(null, "
                    Informe seu Nome.", "Nome");
93              if (cliente1.nome  == null){/**caso o usuário cancele a
                    opção*/
94                  JOptionPane.showMessageDialog(null, "Você cancelou a
                        operação");
95              }
96          }
97          catch (NullPointerException e){
98              JOptionPane.showMessageDialog(null, "Entre com um Nome
                    válido.","Erro" , JOptionPane.ERROR_MESSAGE);
99          }
100
101         try {/** Solicita o sobrenome do cliente e verifica possiveis
                erros */
102             cliente1.sobrenome = JOptionPane.showInputDialog(null, "
                    Informe seu Sobrenome.","Sobrenome");
103         }
104         catch(NullPointerException e){
105             JOptionPane.showMessageDialog(null, "Entre com um
                    sobrenome válido.","Erro" , JOptionPane.ERROR_MESSAGE)
                    ;
106         }
107
108         try{/** Solicita o cpf do cliente e verifica possiveis erros
                */
109             cliente1.cpf = JOptionPane.showInputDialog(null, "Informe
                    o numéro do CPF.","000.000.000-00");
```

```
110         }
111         catch (NullPointerException e){
112             JOptionPane.showMessageDialog(null, "Entre com um CPF
                válido.","Erro" , JOptionPane.ERROR_MESSAGE);
113         }
114         //cliente1.saldo = 445;//remover
115         while(true){ //InterruptedIOException
116 
117             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 deful
118             //conversão de String para int
119             int control = 0;
120             if ("opcao" == null){
121 
122                 control = 0;
123             }
124             else {
125                  control = Integer.parseInt(opcao);
126             }
127 
128             //int control = Integer.parseInt(opcao);
129 
130             if ("opcao"  == null){//caso o usuario cancele a opção
131                 JOptionPane.showMessageDialog(null, "Você cancelou a
                    operação");
132                 break;
133             }
134 
135             if (control == 0){
136                 JOptionPane.showMessageDialog(null, "Você cancelou a
                    operação");
137                 break;
138             }
139 
140             if (control == 4){//para sair da operação
141                 JOptionPane.showMessageDialog(null,"Volte sempre "+
                    cliente1.nome+" "+cliente1.sobrenome+"\nTenha um
                    bom dia!", "LOGOUT", JOptionPane.
                    INFORMATION_MESSAGE, icon);//add custon icon
```

```
142                 break;
143             }
144 
145             switch (control){
146                 case 0:
147                     JOptionPane.showMessageDialog(null, "Você
                           cancelou a operação");
148                     break;
149 
150                 case 1://consulta de saldo   consultaPilas(nome)
151                     JOptionPane.showMessageDialog(null,"Consulta
                           saldo.", cliente1.nome+" "+cliente1.sobrenome,
                            JOptionPane.INFORMATION_MESSAGE, icon);//add
                           custon icon
152                     cliente1.consultaPila(cliente1.nome);
153                     break;
154 
155                 case 2://realiza deposito
156                     JOptionPane.showMessageDialog(null,"Realizar um
                           deposito.", cliente1.nome+" "+cliente1.
                           sobrenome, JOptionPane.INFORMATION_MESSAGE,
                           icon);
157                     int verifica = cliente1.depositaPila(cliente1.
                           nome);
158                     if (verifica == 1){
159                         JOptionPane.showMessageDialog(null,"Depósito
                               realizado com sucesso.", cliente1.nome+" "
                               +cliente1.sobrenome, JOptionPane.
                               INFORMATION_MESSAGE, icon);
160                     }else{
161                         JOptionPane.showMessageDialog(null,"Falha no
                               deposito.\n Tente novamente.", cliente1.
                               nome+" "+cliente1.sobrenome, JOptionPane.
                               ERROR_MESSAGE, icon);
162                     }
163                     break;
164 
165                 case 3:// realiza saque
166                     JOptionPane.showMessageDialog(null,"Realiza um
                           saque.", cliente1.nome+" "+cliente1.sobrenome,
                            JOptionPane.INFORMATION_MESSAGE, icon);
```

```
167                     verifica = cliente1.saquePila(cliente1.nome);
168
169                     if (verifica == 1){
170                         JOptionPane.showMessageDialog(null,"Saque
                               realizado com sucesso.", cliente1.nome+" "
                               +cliente1.sobrenome, JOptionPane.
                               INFORMATION_MESSAGE, icon);
171                     }else{
172                         JOptionPane.showMessageDialog(null,"Falha no
                               saque.\n Tente novamente.", cliente1.nome+
                               " "+cliente1.sobrenome, JOptionPane.
                               ERROR_MESSAGE, icon);
173                     }
174                     break;
175
176                 default:
177                     JOptionPane.showMessageDialog(null,"Opção
                           inválida.", cliente1.nome+" "+cliente1.
                           sobrenome, JOptionPane.INFORMATION_MESSAGE,
                           icon);
178                     break;
179             }
180         }
181     }
182 }
```

Listing 5. *gerenciaBanco.java*