CENTRO UNIVERSITÁRIO CENAC

SANTO AMARO

Sistema

De

atendimento de um laboratório de coleta de sangue

• BRUNO HENRIQUE GOUVEIA DA SILVA

CURO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

São Paulo

2024

Descrição que acontece no sistema implementado:

1. O programa inicia e apresenta um menu de opções para o usuário através de uma caixa de diálogo (**JOptionPane**).
2. O usuário seleciona uma opção do menu, que pode ser:
    - Solicitar nova senha
    - Excluir uma senha
    - Listar todas as senhas
    - Visualizar que é o próximo a ser atendido
    - Chamar o próximo a ser atendido
    - Finalizar Programa
3. Dependendo da opção selecionada, o programa realiza a ação correspondente:
    - Se o usuário selecionar "Solicitar nova senha", o programa pede ao usuário que escolha o tipo de senha (comum ou prioritária) e, em seguida, gera uma nova senha e a adiciona à fila.
    - Se o usuário selecionar "Excluir uma senha", o programa pede ao usuário que informe o nome da senha a ser excluída e, em seguida, remove a senha da fila se ela existir.
    - Se o usuário selecionar "Listar todas as senhas", o programa lista todas as senhas presentes na fila.
    - Se o usuário selecionar "Visualizar que é o próximo a ser atendido", o programa mostra a próxima senha a ser chamada.
    - Se o usuário selecionar "Chamar o próximo a ser atendido", o programa remove a próxima senha da fila e a considera como chamada.
    - Se o usuário selecionar "Finalizar Programa", o programa termina e fecha a caixa de diálogo.
4. O programa continua a apresentar o menu de opções até que o usuário selecione a opção "Finalizar Programa".

Class Teste

```
package ADO2;

import javax.swing.JOptionPane;

/**

 * @author bruno.hgsilva3

 */

public class Teste {

    public static void main(String[] args) {

        Fila<String> fila = new Fila<>();

        String op[] = {"Solicitar nova senha", "Excluir uma senha", "Listar todas as senhas", "Visualizar que é o próximo a ser atendido", "chamar o próximo a ser atendido", "Finalizar Programa"};

        String operacao = "";

        do {

            operacao = (String) JOptionPane.showInputDialog(null, "Informe operação desejada:", "Opçao",

                    JOptionPane.INFORMATION_MESSAGE, null, op, op[0]);

            switch (operacao) {

                case "Finalizar Programa" -> {

                    JOptionPane.showMessageDialog(null, "Ate a Proxima!!!");

                }

                case "Solicitar nova senha" -> {

                    Object escolha = JOptionPane.showInputDialog(null, "Qual tipo de senha:", "Opção", JOptionPane.INFORMATION_MESSAGE, null, fila.tipo, fila.tipo[0]);

                    if (escolha.equals("Prioridade")) {

                        fila.setPreferencia(true);

                    }

                    fila.gerarSenha();

                }

                case "Excluir uma senha" -> {

                    String nome = JOptionPane.showInputDialog("Informe o nome da senha:");

                    if (JOptionPane.showConfirmDialog(null, "Deseja remove essa senha") == 0) {

                        try {

                            fila.remove(fila.buscaPosicao(nome));

                            JOptionPane.showMessageDialog(null, "Senha excluida com sucesso");

                        } catch (Exception e) {

                            JOptionPane.showMessageDialog(null, e);

                        }

                    }

                }
```

```
case "Listar todas as senhas" -> {
                if (fila.estaVazia()) {
                    JOptionPane.showMessageDialog(null, "Fila esta vazia");
                } else {
                    JOptionPane.showMessageDialog(null, fila);
                }
            }
            case "Visualizar que é o próximo a ser atendido" -> {
                if (fila.estaVazia()) {
                    JOptionPane.showMessageDialog(null, "Fila esta vazia");
                } else {
                    JOptionPane.showMessageDialog(null, "Proximo a senha a ser chamada é: " + fila.espiar());
                }
            }
            case "chamar o próximo a ser atendido" -> {
                if (fila.estaVazia()) {
                    JOptionPane.showMessageDialog(null, "Fila esta vazia");
                } else {
                    fila.chamarProximo();
                }
            }
            default ->
                JOptionPane.showMessageDialog(null, "opção invalida, tente novamento");
        }
    } while (!operacao.equalsIgnoreCase("Finalizar Programa"));
  }
}
```

Class Fila

```
import javax.swing.JOptionPane;
/** @author bruno.hgsilva3*/
public class Fila<T> extends EstruturaEstatica<T> {
    final String tipo[] = {"Comum", "Prioridade"};
    public int numComun = 1;
    public int numPrioridade = 1;
    public int p = 0;
    public Fila(int capacidade) {
        super(capacidade); }
    public Fila() {
        super();  }
    //metodo para gerar senha
    public void gerarSenha() {
        Object senha = "";
        if (isPreferencia()) {
            senha = "P" + this.numPrioridade;
            this.numPrioridade++;
        } else {
            senha = "C" + this.numComun;
            this.numComun++;    }
        if (adiciona((T) senha)) {
            JOptionPane.showMessageDialog(null, "Senha gerada com sucesso\n Senha: " + senha);
        } else {
            JOptionPane.showMessageDialog(null, "errro a gerar a senha"); }   }
    public T espiar() {
        if (super.elementos[0].toString().contains("P") && this.p < 3) {
            return this.elementos[0];
        } else {
            if (super.elementos[0].toString().contains("C")) {
                return this.elementos[0];
            } else if (p == 3) {
                for (int i = 0; i < elementos.length; i++) {
                    if (super.elementos[i].toString().contains("C")) {
                        return this.elementos[i];
                    } }   } }
        return null;
    }
```

```
public void chamarProximo() {
        if (super.elementos[0].toString().contains("P") && this.p < 3) {
            try {
                JOptionPane.showMessageDialog(null, "Senha chamada: " + super.elementos[0].toString());
                remove(0);
                this.p++;
            } catch (Exception e) {
                JOptionPane.showMessageDialog(null, e);
            }
        } else {
            if (super.elementos[0].toString().contains("C")) {
                try {
                    JOptionPane.showMessageDialog(null, "Senha chamada: " + super.elementos[0].toString());
                    remove(0);
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(null, e);
                }
            } else if (p == 3) {
                for (int i = 0; i < elementos.length; i++) {
                    if (super.elementos[i].toString().contains("C")) {
                        try {
                            JOptionPane.showMessageDialog(null, "Senha chamada: " + super.elementos[i].toString());
                            remove(i);
                            break;
                        } catch (Exception e) {
                            JOptionPane.showMessageDialog(null, e);
                        }
                    }
                }
            }
            p = 0;
        }
    }
}
```

Class EstruturaEstatica

```
package ADO2;

/** @author bruno.hgsilva3 */

public class EstruturaEstatica<T> {

    public T[] elementos;

    public int tamanho;

    private boolean preferencia = false;

    public int qtdPreferencia = 0;

    public boolean isPreferencia() {

        return preferencia; }

    public void setPreferencia(boolean preferencia) {

        this.preferencia = preferencia; }

    //metodo construtor com paramentro

    public EstruturaEstatica(int capacidade) {

        this.elementos = (T[]) new Object[capacidade];

        this.tamanho = 0; }

    // metodo construtor sem parametro

    public EstruturaEstatica() {

        this(10);}

// metodo para adicionar elementos

    public boolean adiciona(T elemento) {

        this.aumentaCapacidade();

        if (this.tamanho < this.elementos.length) {

            if (this.isPreferencia()) {

                int k = tamanho;

                while (k > qtdPreferencia) {

                    this.elementos[k] = this.elementos[k - 1];

                    k--; }

                this.elementos[k] = elemento;

                this.tamanho++;

                qtdPreferencia++;

                setPreferencia(false);

                return true; }

            this.elementos[this.tamanho] = elemento;

            this.tamanho++;

            return true; }

        return false;

    }
```

```
//metodo para aumentar a capacidade
    public void aumentaCapacidade() {
        if (this.elementos.length == this.tamanho) {
            T[] elementosNovos = (T[]) new Object[this.elementos.length * 2];
            for (int i = 0; i < this.elementos.length; i++) {
                elementosNovos[i] = this.elementos[i];  }
            this.elementos = elementosNovos;}   }
    public int tamanho() {
        return this.tamanho; }
    @Override
    public String toString() {
        StringBuilder s = new StringBuilder();
        s.append("[");
        for (int i = 0; i < this.tamanho - 1; i++) {
            s.append(this.elementos[i]);
            s.append(", "); }
        if (this.tamanho > 0) {
            s.append(this.elementos[this.tamanho - 1]);  }
        s.append("]");
        return s.toString(); }
    public boolean estaVazia() {
        return this.tamanho == 0;  }
    public void remove(int posicao) throws Exception {
        if (posicao >= 0 && posicao < this.tamanho) {
            if (elementos[posicao].toString().contains("P")) {
                qtdPreferencia--; }
            for (int i = posicao; i < this.tamanho - 1; i++) {
                this.elementos[i] = this.elementos[i + 1]; }
            this.tamanho--;
        } else {  throw new Exception("Senha Invalida");  }}
    public int buscaPosicao(T elemento) {
        for (int i = 0; i < tamanho; i++) {
            if (this.elementos[i].equals(elemento)) {
                return i;  } }
      return -1; }
}
```