UNIVERSIDADE DO OESTE PAULISTA
UNOESTE
PRESIDENTE PRUDENTE - SP
Campus I Rua José Bongiovani, 700
Cidade Universitária CEP 19050 900
Fone(18) 3229 1000

Faculdade de Informática de Presidente Prudente
Fone (18) 3229 1060
www.unoeste.br/fipp
e-mail: fipp@fipp.unoeste.br

FIPP
FACULDADE DE INFORMÁTICA
DE PRESIDENTE PRUDENTE
CAMPUS I

# Pesquisa e Ordenação

Prof. Francisco Assis da Silva

– Atividade Prática de Implementação:
Animação de dois Métodos de Ordenação –

Fazer um programa em JavaFX para realizar a animação intuitiva de dois dos algoritmos de ordenação estudados (e ou não estudados em sala de aula). Os algoritmos que você deverá fazer a animação são numerados e descritos na sequência. Observe que os dois últimos números do seu RA serão utilizados na escolha de quais métodos de ordenação você deverá implementar na animação.

0 – Heap Sort
1 – Shell Sort
2 – Quick Sort
3 – Merge Sort (1ª Implementação)
4 – Counting Sort
5 – Bucket Sort
6 – Radix Sort
7 – Comb Sort
8 – Gnome Sort
9 – Tim Sort

**Obs**: se você tiver os dois últimos números do RA iguais, pegue um deles e some 1, ou, no caso de ser dígito 9, então o método passa a ser o de número 0.

A animação, para cada algoritmo, deverá conter:

1. geração aleatória dos valores a serem ordenados;
2. objetos com os valores sendo movidos na tela;
3. código-fonte do método de ordenação demarcando cada linha de execução;
4. variáveis de índices mostradas na tela, setas indicando o processo, cores, etc.

Para se ter ideia de como mover dois objetos Button na tela simulando uma permutação, segue o código em JavaFX de exemplo:

```
package src;

import javafx.application.Application;
import javafx.application.Platform;
import javafx.concurrent.Task;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;
import javafx.scene.text.Font;
```

```
public class Principal extends Application {

    AnchorPane pane;
    Button botao_inicio;
    private Button vet[];

    public static void main(String[] args)
    {
        launch(args);
    }

    @Override
    public void start(Stage stage) throws Exception
    {
        stage.setTitle("Pesquisa e Ordenacao");
        pane = new AnchorPane();

        botao_inicio = new Button();
        botao_inicio.setLayoutX(10); botao_inicio.setLayoutY(100);
        botao_inicio.setText("Inicia...");
        botao_inicio.setOnAction(e->{ move_botoes();});
        pane.getChildren().add(botao_inicio);

        vet = new Button[2];
        vet[0] = new Button("10");
        vet[0].setLayoutX(100);   vet[0].setLayoutY(200);
        vet[0].setMinHeight(40);  vet[0].setMinWidth(40);
        vet[0].setFont(new Font(14));
        pane.getChildren().add(vet[0]);

        vet[1] = new Button("20");
        vet[1].setLayoutX(180);   vet[1].setLayoutY(200);
        vet[1].setMinHeight(40);  vet[1].setMinWidth(40);
        vet[1].setFont(new Font(14));
        pane.getChildren().add(vet[1]);

        Scene scene = new Scene(pane, 800, 600);
        stage.setScene(scene);
        stage.show();
    }

    public void move_botoes()
    {
        Task<Void> task = new Task<Void>(){
            @Override
            protected Void call() {

                //permutação na tela
                for (int i = 0; i < 10; i++) {
                    Platform.runLater(() -> vet[0].setLayoutY(vet[0].getLayoutY() + 5));
                    Platform.runLater(() -> vet[1].setLayoutY(vet[1].getLayoutY() - 5));
                    try {
                        Thread.sleep(50);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }

                for (int i = 0; i < 16; i++) {
                    Platform.runLater(() -> vet[0].setLayoutX(vet[0].getLayoutX() + 5));
```

```
                    Platform.runLater(() -> vet[1].setLayoutX(vet[1].getLayoutX() - 5));
                    try {
                        Thread.sleep(50);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }

                for (int i = 0; i < 10; i++) {
                    Platform.runLater(() -> vet[0].setLayoutY(vet[0].getLayoutY() - 5));
                    Platform.runLater(() -> vet[1].setLayoutY(vet[1].getLayoutY() + 5));
                    try {
                        Thread.sleep(50);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }

                //permutação na memória
                Button aux = vet[0];
                vet[0] = vet[1];
                vet[1] = aux;

                return null;
            }
        };
        Thread thread = new Thread(task);
        thread.start();
    }
}
```