

- Projeto da Disciplina: Algoritmo e Estrutura de Dados I
- Tema do Projeto: Sistema de Priorização de Tarefas em Tempo Real

Autores:

Josiely Toledo - 02220356

Laura Trigo - 02210361

Rafael Souza - 02220346

# Introdução

- Este projeto simula um Sistema de Priorização de Tarefas (SPT) usado para organizar tickets de suporte em uma empresa de desenvolvimento.
- Cada tarefa possui prioridade de 1 a 10, sendo 1 a mais crítica. O backlog deve permanecer sempre ordenado, inserindo cada nova tarefa na posição correta.
- Para atender ao enunciado, utilizei as estruturas sequenciais apresentadas em aula **Lista** e **Fila** aplicando sobre elas a lógica de inserção do algoritmo **Insertion Sort** para manter a ordenação em tempo real.

# Objetivo

- Manter o backlog de tarefas sempre ordenado por prioridade.
- Prioridade variando entre **1 (mais crítica)** e **10 (menos urgente)**.
- Inserir cada nova tarefa na **posição correta** dentro da lista.
- Aplicar o algoritmo **Insertion Sort** durante o processo de inserção.
- Utilizar as estruturas **Lista** e **Fila** apresentadas em aula como base da solução.
- Garantir que a lista seja atualizada automaticamente em tempo real a cada nova inserção.

## Técnica Utilizada

As classes utilizadas foram:

- **Dados** – representa cada tarefa (ID e prioridade).
- **Lista** – estrutura sequencial usada para armazenar o backlog.
- **Fila** – estrutura auxiliar usada para deslocamento de elementos.
- **BacklogFila** – aplica a lógica de inserção ordenada (Insertion Sort).
- **JanelaSPT** – interface gráfica para interação com o usuário.

O sistema foi desenvolvido de forma modular, separando cada responsabilidade em uma classe específica.

# Classe Dados

- Representa cada tarefa do backlog.
- Contém os atributos:
  - id (identificador da tarefa)
  - prioridade (1 a 10)
- Possui métodos *getters* para acessar os valores.
- Usada como unidade básica dentro da Lista e da Fila.
- Serve como modelo de tarefa utilizado em toda a lógica de ordenação.

# Classe Lista

- Estrutura de dados **sequencial** baseada em vetor dinâmico (Object[]).
- Permite armazenar, acessar, inserir e remover elementos.
- Redimensiona o vetor automaticamente quando necessário.
- Métodos principais:
  - adiciona(objeto)
  - adiciona(posicao, objeto)
  - pega(posicao)
  - remove(posicao)
  - vazia()
- Utilizada como base para manter o backlog ordenado.

# Classe Fila

- Implementada utilizando a classe Lista.
- Segue o princípio FIFO (*First In, First Out*).
- Opera inserções no final e remoções no início.
- Métodos principais:
  - Insere(objeto) – adiciona ao final da fila
  - remover() – remove o primeiro elemento
  - pEmpty() – verifica se a fila está vazia
  - mostra() – exibe o conteúdo da fila
- Utilizada como **estrutura auxiliar** para deslocamento temporário de elementos durante a inserção ordenada.

# Classe BacklogFila

- Mantém o backlog sempre **ordenado por prioridade**.
- Implementa a lógica do **Insertion Sort** na inserção.
- Métodos principais:
  - Comparar da **direita para a esquerda**.
  - Identificar onde a nova tarefa deve entrar.
  - Usar a **Fila** para guardar temporariamente elementos que serão deslocados.
  - Reposicionar os elementos e inserir a nova tarefa.
  - Lista → armazenamento do backlog.
  - Fila → estrutura auxiliar usada para deslocar elementos.
- É a classe responsável pelo funcionamento central do sistema de Ordenação.



# Insertion Sort

- Percorre o backlog **da direita para a esquerda**.
- Compara a **nova tarefa** com as tarefas existentes.
- Identifica a **posição correta** onde ela deve entrar.
- Tarefas com prioridade **maior** são deslocadas temporariamente.
- Fila auxiliar **armazena** os elementos deslocados.
- Após abrir espaço, a nova tarefa é **inserida**.
- Os elementos são **recolocados** na ordem correta.

# Interface Gráfica

- Permite inserir ID e prioridade da tarefa.
- Botão “**Adicionar Tarefa**” para registrar nova entrada.
- **Atualização automática** do backlog em tempo real.
- Exibe a **lista ordenada** após cada inserção.
- Realiza validações de entrada (valores numéricos, faixa 1–10).
- Desenvolvida com Swing (JFrame, JTextField, JButton, JTextArea).

# Análise

- O sistema mantém o backlog sempre ordenado por prioridade.
- A inserção utiliza o princípio do Insertion Sort para manter a consistência da lista.
- A interface gráfica proporciona clareza, usabilidade e fácil validação.
- A solução reflete o comportamento de um sistema real de atendimento e suporte.
- A implementação integra teoria e prática, mostrando o funcionamento real de uma fila de priorização.