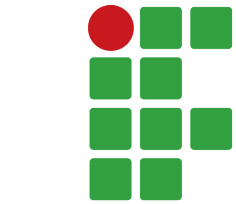

INSTITUTO
FEDERAL
Santa Catarina

# Relatório final

*Git e Robocode*

Alunos: Eduardo C. Oliveira e Talles S. da Cruz, 2025
Professor: Diego da Silva de Medeiros

# 1 Introdução

Nesta atividade, exploramos um pouco do Robocode, uma plataforma de programação de batalhas de robôs para a prática da lógica de programação em JAVA. O principal objetivo desta atividade foi construir um código de robô e utilizar na prática o Git e o GitHub para controle de versão do mesmo.

A importância deste controle de versão no desenvolvimento é fundamental, pois ela permite que possamos observar todas as alterações do código feitas por ambos os colaboradores, ver o que e quando cada um editou algo, e também nos permite, caso necessário, reverter as ações feitas para versões anteriores.

# 2 Objetivos da atividade

Os objetivos propostos com esta atividade foram:

- Utilizar o Robocode como ferramenta de aprendizagem;
- Compreender o funcionamento do Git, conhecendo comandos para o controle de versão;
- Entender o conceito de versionamento de código;
- Aplicar o GitHub como plataforma de colaboração entre a dupla;
- Desenvolver o trabalho em equipe, comunicação e resolução de problemas.

# 3 Descrição da atividade

A atividade foi realizada com o objetivo principal de explorar a plataforma Robocode para a prática da lógica de programação em JAVA e, simultaneamente, utilizar o Git e o GitHub para o controle de versão do código do robô. O trabalho se voltou ao desenvolvimento do robô que chamamos de Shigeo Kageyama.

Para começar a programar nosso robô, inicialmente buscamos referências na internet, através da Wiki do Robocode (que pode ser acessada por **este link**) e de alguns repositórios públicos. Depois de estudarmos alguns robôs, observando suas estratégias, funcionalidades e código, partimos para a etapa de planejar a estratégia do nosso robô. Todo o planejamento da estratégia foi discutido entre os integrantes do grupo e devidamente documentado, para que não perdêssemos nenhuma ideia de futuras implementações.

Assim, começamos a desenvolver nosso projeto, nos baseando em estratégias discutidas e em dois robôs que, inicialmente, consideramos muito bons: o **Omni Bot** e o robô **BarbieScript**. A partir desse momento, utilizamos diversas ferramentas (Wiki do Robocode, manual disponível no SIGAA, ChatGPT, Gemini AI e repositórios do GitHub) para criarmos a primeira versão do nosso robô.

A utilização de inteligência artificial foi muito útil, pois enviamos algumas ideias de estratégias nos chats para tentar entender como funcionariam e como poderíamos aplicá-las em nosso robô. Em algumas ocasiões, as IAs mostravam um modelo do que alterar ou uma implementação direta no código do que foi solicitado. É importante notar que, na maioria dos casos, as sugestões que as IAs enviavam prontas para rodar não funcionavam do jeito que esperávamos, ou então apagavam algo já existente.

Durante todo o desenvolvimento do robô, utilizamos um repositório que foi criado no GitHub para armazenar as diversas alterações e versões do nosso robô. Sempre que um dos integrantes fazia alguma alteração, o mesmo utilizava o Git para enviar a nova versão ao repositório, mantendo-o sempre atualizado. Também tivemos o cuidado de enviar o código apenas se ele estivesse funcionando, para evitar quaisquer problemas futuros (como esquecer de corrigir um bug ou acontecer algum conflito no código, por exemplo). Para cada alteração importante, enviávamos um commit com uma breve descrição do que foi alterado no código e o que foi adicionado de novo, para assim termos uma noção e organização do andamento do projeto.

Por fim, pouco antes das batalhas em sala de aula, também utilizamos os repositórios dos outros alunos para consulta e desenvolvimento do nosso robô, a fim de realizar testes entre o nosso e os demais robôs e, com isso, aprimorar o código do nosso robô e tentar vencer a batalha final. Todos os repositórios que foram consultados eram públicos.

# 4 Estrutura do Git utilizada

Como visto anteriormente, para a construção do robô Shigeo Kageyama, foi criado um repositório no GitHub que, ao longo do tempo, passou a ter diversas branches. Cada branch continha alterações no código do robô feitas por um dos integrantes da dupla.

A branch Main foi definida como a branch principal, contendo o código do robô sempre atualizado e funcionando com todas as funcionalidades criadas nas demais branches, que serão detalhadas a seguir (elas não estão em ordem cronológica de modificação):

- **Branch Main:** A branch principal do projeto, que recebeu merges de todas as outras branches, exceto a de inspirações. Nela, foi adicionada a apresentação do robô e, em breve, este relatório também será incluído.
- **Branch Colisões:** Esta branch introduziu no robô o sistema de detecção de colisões contra paredes, tiros e robôs inimigos. Um código para movimentação frontal do robô também foi adicionado para testar a funcionalidade das funções implementadas.
- **Branch Comemoração_de_vitória:** Ao final do código, foi adicionada a "dança da vitória". Toda vez que o robô ganhava um round, ele dançava e trocava de cor aleatoriamente por um período de tempo.
- **Branch Distancia_segura:** Esta branch implementou a parte do código que faz com que o robô se mantenha a uma distância segura dos demais. Ela adicionou uma constante para isso e realizou algumas alterações no cálculo adaptativo de força, que foi introduzido em outra branch.
- **Branch Implementações_finais:** Esta branch foi utilizada para criar uma versão aprimorada do robô que já havíamos finalizado e denominado como versão 1.0, adicionando novas edições e melhorias ao código.
- **Branch Inspirações:** Nesta branch, foram adicionados arquivos de outros robôs que utilizamos como ferramenta de estudo e inspiração para a criação do Shigeo.
- **Branch Mira:** Nesta branch, implementamos o sistema de mira do robô e o aprimoramos para que ficasse mais preciso e eficiente.
- **Branch Movimentação:** Nesta branch, o modo de movimentação do robô foi adicionado, fazendo com que ele se movimente perpendicularmente ao inimigo.
- **Branch Tiro:** Por fim, esta branch trouxe o sistema de tiro do robô e o sistema de mudança da força do tiro por proximidade, tornando-o mais eficiente.

## Commits e Pull Requests

Quanto aos commits feitos durante o desenvolvimento, eles mantiveram, na maioria das vezes, um padrão de adicionar como descrição do commit o que foi feito naquela versão, explicando todas as alterações. O nome do commit, por muitas vezes, seguiu o padrão do próprio Git (`Update`, `Merge`, `Add`, etc.).

Neste projeto, tivemos algumas pull requests. Em resumo, as pull requests são uma maneira de notificar outros membros da equipe que você concluiu um conjunto de alterações em uma branch própria e gostaria que essas alterações fossem revisadas para

que, posteriormente, fossem adicionadas ao código principal. Fizemos uso das pull requests em cada uma das branches que criamos. Após realizar as alterações desejadas no robô, criávamos um pull request, revisávamos o código e adicionávamos o novo conteúdo ao código principal.

# 5 Aprendizados e Considerações Finais

Ao longo da atividade de desenvolvimento do robô Shigeo Kageyama, adquirimos conhecimentos significativos sobre o controle de versão de códigos e aprimoramos consideravelmente nosso trabalho em equipe. A colaboração foi fundamental em todas as etapas do projeto, desde a concepção das estratégias de programação do robô até a estruturação do repositório no GitHub e a definição das atribuições de cada integrante. Essas discussões e decisões conjuntas foram essenciais para o sucesso do projeto e representam um aprendizado valioso para futuras iniciativas e para a nossa vida profissional.

## A Importância do Git no Desenvolvimento

O uso do Git e do GitHub se mostrou uma ferramenta de extrema importância e utilidade. Esse contato inicial permitiu com que nos familiarizássemos com a ferramenta, que é indispensável no desenvolvimento de software moderno. A prática constante com o Git solidificou nosso entendimento sobre conceitos como branches, merges, commits e pull requests, coisas que serão empregadas ao longo de toda a nossa carreira profissional. A experiência prática com o controle de versão nos proporcionou uma base para gerenciar projetos e colaborar de forma eficiente em equipes.

## Benefícios da Atividade

A atividade foi de grande importância por diversos motivos, como por exemplo:

- **Desenvolvimento de Habilidades Colaborativas:** A necessidade de trabalhar em dupla para resolver problemas complexos do projeto estimulou a comunicação eficaz, a divisão de tarefas e a capacidade de conciliar diferentes ideias e abordagens.
- **Gerenciamento de Projetos e Organização:** A estruturação do repositório no GitHub e o uso do controle de versão nos ensinaram a organizar o código de forma lógica e a manter um histórico claro das alterações, aspectos muito importantes para a manutenção e evolução de qualquer projeto de software.
- **Resolução de Problemas e Pensamento Crítico:** Os desafios de criar um robô, pensando em toda a sua estratégia e em problemas que ele poderia sofrer contra inimigos exigiu a capacidade de propor soluções criativas para todas estas questões e também bastante diálogo.
- **Aplicação Prática de Conhecimentos Teóricos:** De certa forma, a atividade proporcionou uma chance de aplicar conhecimentos de programação e lógica vistos tanto nesta disciplina quanto na disciplina de Pensamento Computacional e Algorítimos, proporcionando um melhor aprendizado.
- **Incentivo à Autonomia e Pesquisa:** Muitas das funcionalidades implementadas exigiram pesquisa e aprendizado autônomo, incentivando nossa capacidade de buscar conhecimento e resolver problemas por conta própria.

# 6 Anexos

Abaixo, segue uma foto de como ficou a aparência do nosso robô, uma captura de tela da Branch Main do nosso repositório no Github e o link do mesmo.


Figura 1: Robô em batalha

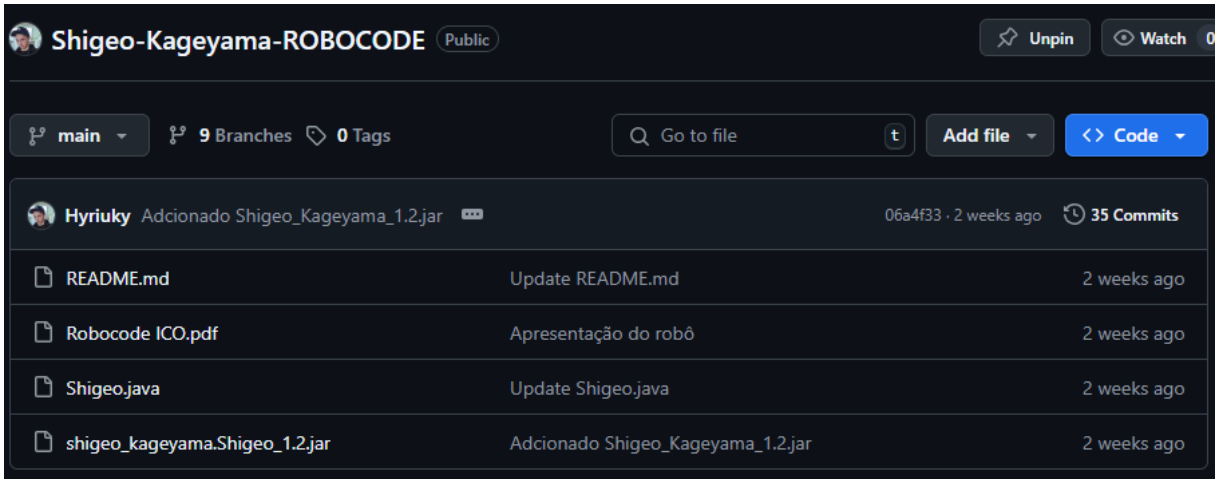


Figura 2: Print do repositório no GitHub. **Acesse ele aqui.**