RELATÓRIO TÉCNICO: Implementação do jogo “Ligue 4” em linguagem C

Universidade Federal de Sergipe (UFS)
Departamento de Computação Disciplina: Programação Imperativa
Docente: Prof. Adam Lucas Pinheiro da Silva
Discentes: Arthur Nicolas Silva Lima, Lauane de Morais Araujo, Luiza Pereira Accioly

1. Arquitetura de Dados e Mecânica de Jogo

O projeto foi estruturado de forma **modular**, separando a interface (display.c), a lógica de controle (verificarcolunas.c) e os motores de decisão (vitoria.c e módulos de IA).

- **Representação do Tabuleiro:** Utilizou-se uma matriz estática bidimensional int tabuleiro[6][7]. O estado de cada célula é definido por constantes inteiras: 0 (vazio), 1 (Jogador 1) e 2 (Jogador 2).
- **Mecânica de Gravidade:** Diferente de matrizes convencionais, a inserção no Ligue 4 é restrita pela base. Implementou-se um algoritmo de busca que recebe o índice da coluna e itera da linha 5 à 0. A peça é alocada na primeira célula cujo valor seja 0, simulando o empilhamento físico.

2. Algoritmos de Detecção (Vitória e Empate)

A verificação de estado final ocorre após cada movimento:

- **Varredura por Janelas:** Para vitória horizontal, vertical e diagonais, o algoritmo utiliza uma "janela deslizante" de 4 posições. Ele percorre a matriz verificando se a soma de peças idênticas em uma sequência linear atinge o objetivo, respeitando rigorosamente os limites dos índices para evitar *Buffer Overflow*.
- **Tratamento de Diagonais:** Foram implementados dois laços distintos: um para a diagonal principal (descendente: i+k, j+k) e outro para a secundária (ascendente: i-k, j+k).
- **Condição de Empate:** O sistema monitora o número total de jogadas. Caso atinja o limite de 42 inserções (capacidade total da matriz) sem que um vencedor seja declarado, o jogo encerra o loop principal retornando o estado de empate.

3. Inteligência Artificial e Decisão

A IA foi projetada para escalar em complexidade, utilizando o conceito de **priorização de estados**:

- **Nível 1 (Baixa):** Utiliza a função rand() da biblioteca stdlib.h, alimentada por srand(time(NULL)) para garantir a imprevisibilidade.
- **Nível 2 (Média):** Implementa uma lógica de **análise preventiva**. A IA simula a jogada do oponente e, caso identifique um "Ligue 3" iminente, utiliza a função

validarGravidade para checar se o bloqueio naquela coluna é válido (ou seja, se a peça cairá exatamente na linha da ameaça).
- **Nível 3 (Alta):** Introduz uma mentalidade **ofensiva-defensiva**. O algoritmo primeiro busca por "janelas de vitória" próprias para encerrar a partida. Caso não encontre, aciona os protocolos de defesa do Nível 2.

4. Distribuição de Tarefas e Metodologia

A equipe utilizou o modelo de desenvolvimento modular para permitir o trabalho paralelo:

- **Arthur Nicolas:** Desenvolveu o *core* do sistema (motor de jogo), loop de turnos, lógica de gravidade, verificadores de vitória em eixos fixos (H/V) e a base da IA randômica.
- **Lauane Morais:** Responsável pela experiência do usuário (UX), implementando a interface via terminal com cores ANSI, menus de navegação, documentação técnica, o algoritmo de defesa da IA Nível 2 e tratamento de erros.
- **Luiza Accioly:** Especializou a lógica geométrica do tabuleiro, desenvolvendo os algoritmos de verificação diagonal e a IA Nível 3, focada em estratégias de ataque e otimização de busca.

5. Dificuldades Técnicas e Soluções

O principal obstáculo foi a **modularização com múltiplos arquivos e lógica de diagonais e construção de Inteligência Artificial II e III**. Erros de "multiple definition" surgiram ao declarar variáveis globais incorretamente. A solução consistiu na aplicação de *Header Guards* e no uso de cabeçalhos apenas para protótipos de funções, assim como o impasse das lógicas mais complexas do projeto, desde lógicas de estruturação do tabuleiro até as IA's. Além disso, a automação via **Makefile** foi crucial para gerenciar a compilação de múltiplos módulos .c de forma eficiente.