Renan Balbino de Medeiros

Prof. Fernando Marques Figueira Filho

Introdução às Técnicas de Programação

25 de setembro de 2025

Relatório técnico

Este projeto tem como objetivo o desenvolvimento de um jogo **Campo Minado** na linguagem de programação **C**, empregando as técnicas e conceitos aprendidos ao longo da disciplina. A escolha do jogo se justifica por sua relevância como ferramenta de entretenimento e, ao mesmo tempo, pelo desafio técnico que apresenta desde as etapas iniciais, especialmente na implementação de **vetores** e em seu adequado mapeamento, propostos como entrega parcial do trabalho.

Além do aspecto técnico, o projeto também busca aliar prática e motivação, oferecendo uma abordagem pedagógica que valoriza o aprendizado por meio da experimentação e do caráter lúdico do jogo. Essa combinação possibilitou observar, na prática, a aplicação dos conteúdos estudados e explorar de forma produtiva os benefícios de aprender enquanto se desenvolve uma solução interativa.

ANÁLISE TÉCNICA

Nesta seção, apresenta-se a análise técnica do projeto desenvolvido, detalhando a metodologia adotada, as ferramentas utilizadas, a aplicação dos conceitos trabalhados na disciplina e a organização do código. O objetivo é evidenciar como os fundamentos teóricos foram aplicados

de forma prática na construção do jogo **Campo Minado**, destacando tanto as escolhas tecnológicas quanto às estratégias de implementação.

Metodologia.

O desenvolvimento foi conduzido em ambiente Linux, considerando a robustez e a compatibilidade desse sistema operacional para programação em C. As principais ferramentas empregadas foram:

- Compilador: GCC (GNU Compiler Collection), padrão em ambientes Linux para compilação de código C.
- Editor: Visual Studio Code, um editor de código-fonte moderno com suporte a extensões, depuração e integração com terminal.
- Sistema Operacional: Linux, que oferece ambiente robusto para desenvolvimento em C.
- Bibliotecas: Foram utilizadas as bibliotecas padrão do C: stdio.h, stdlib.h, time.h e ctype.h.

Aplicação dos Conceitos da U1.

O projeto fez uso extensivo de estruturas condicionais (if, else if, else) para controlar a lógica central do jogo. Elas foram fundamentais para validar as posições escolhidas pelo usuário, verificar se a célula continha uma bomba, definir as condições de vitória e derrota, e coordenar o

processo de inicialização e atualização do tabuleiro. Por exemplo:

```
if (tabuleiro[linha][col] == -1) {
    // Encontrou bomba
} else {
    // Casa segura
}
```

As estruturas de repetição (for e while) também desempenharam papel essencial, sendo aplicadas na inicialização e no percorrimento das matrizes que compõem o tabuleiro, na distribuição aleatória das bombas e no cálculo do número de bombas vizinhas a cada célula. Além disso, foram empregadas para verificar continuamente se a condição de vitória havia sido atingida, ou seja, se todas as casas seguras haviam sido abertas, como no exemplo a seguir.

```
for (int i = 0; i < TAM; i++) {
    for (int j = 0; j < TAM; j++) {
        // Operações sobre o tabuleiro
    }
}
```

O uso de matrizes bidimensionais se mostrou indispensável para a representação do tabuleiro. A matriz tabuleiro[TAM][TAM] armazena o estado de cada célula, que pode conter uma bomba (valor -1) ou indicar a quantidade de bombas vizinhas (valores de 0 a 8). Já a matriz visivel[TAM][TAM] registra o estado de visibilidade das casas, distinguindo entre aquelas já reveladas e as ainda ocultas. Essa estrutura de dados permitiu gerenciar de forma eficiente tanto o conteúdo do jogo quanto as interações do jogador.

A organização do código também foi pensada de forma modular, com funções dedicadas a tarefas específicas. A função inicializar() é responsável por preparar o tabuleiro e posicionar as bombas, enquanto imprimir() exibe ao usuário o estado atual do jogo. A função calcular_perigo() atribui a cada célula o número de bombas vizinhas, e venceu() verifica se a condição de vitória foi alcançada. Essa separação em funções contribuiu para a clareza, legibilidade e manutenção do código, além de facilitar futuras expansões.

Estrutura de dados.

O projeto utilizou variáveis auxiliares que desempenharam papéis importantes no controle do jogo. Variáveis como *linha* e *coluna* gerenciaram a entrada do usuário e a posição selecionada, enquanto *ativo* controlou o loop principal e *bombas_colocadas* auxiliou na lógica de distribuição aleatória. Juntas, essas estruturas reforçam a coerência do projeto e demonstram a aplicação prática dos conteúdos estudados.

IMPLEMENTAÇÃO E REFLEXÃO

Durante o desenvolvimento do projeto, algumas dificuldades técnicas se destacaram. A primeira delas foi compreender a lógica necessária para percorrer corretamente as matrizes bidimensionais, o que exigiu o uso de repetições aninhadas. A solução para esse desafio veio ao perceber que cada dimensão da matriz deveria ser controlada por uma estrutura de repetição própria, permitindo o acesso adequado a linhas e colunas. Outro obstáculo foi a implementação da aleatoriedade na posição das bombas, que só pôde ser superado após o estudo da biblioteca time.h, utilizada para garantir que a distribuição das bombas fosse diferente a cada execução do programa. Além disso, houve um dilema em relação à lógica de vitória e derrota do jogo,

solucionado pela adoção de duas matrizes distintas: uma visível ao jogador e outra usada como “gabarito”, que armazena as informações reais do tabuleiro. A função calcular_perigo() contribuiu para esse processo ao contabilizar o número de bombas vizinhas em cada jogada, elevando a qualidade da jogabilidade.

Outro ponto relevante foi a adaptação da forma de exibição do tabuleiro. Inicialmente, linhas e colunas eram representadas apenas por números, mas optou-se por utilizar letras para as colunas e números para as linhas, tornando a interação mais intuitiva. Para isso, foi necessário compreender a relação entre caracteres e seus respectivos códigos ASCII, o que possibilitou a conversão das letras digitadas pelo jogador em índices utilizáveis pela matriz. Além disso, o uso de diretivas como #define mostrou-se valioso, permitindo centralizar variáveis de uso global e simplificar a passagem de parâmetros entre as funções, deixando o código mais limpo e organizado.

A estrutura modular do programa também foi uma decisão importante. A divisão em funções específicas, como as responsáveis por inicializar o tabuleiro, exibir o estado do jogo, calcular o perigo e verificar a vitória, garantiu maior clareza e facilidade de manutenção. Essa escolha favorece ainda a expansão do projeto em etapas futuras, sem comprometer o funcionamento atual.

Em termos de aprendizado, o projeto proporcionou uma vivência prática da aplicação de conceitos fundamentais da programação em C, especialmente no que se refere à manipulação de

vetores bidimensionais, ao controle de fluxo e ao uso de bibliotecas padrão. Como melhorias futuras, planeja-se ampliar a jogabilidade ao implementar recursos típicos de versões mais completas do Campo Minado, como a abertura automática de áreas seguras, o uso de bandeiras para marcar bombas suspeitas, a criação de um menu inicial para escolha de dificuldade, além da possibilidade de salvar e carregar partidas. Tais avanços representariam não apenas um ganho em termos de experiência do jogador, mas também novos desafios técnicos a serem superados, mantendo o caráter educativo e formativo do projeto.