

**UNIVERSITÁRIO AUTÔNOMO DO BRASIL**

**Escola Politécnica – Engenharia de Software**

**APS2 – Atividade Prática Supervisionada**

**Disciplina: Programação Avançada**

Período/Turno/Turma: 5º Período – Noite – 5ES

**Equipe:** Arthur Damiao Mendes

**Professor:** Fábio Garcez Bettio

**CODLEC**

Desenvolvimento de um Jogo de Adivinhação de Palavras em  
Linguagem C para Terminal

Curitiba

2025

# Resumo

Este artigo apresenta o desenvolvimento do CodleC, um jogo de adivinhação de palavras inspirado no popular Wordle, implementado inteiramente em linguagem C para execução em terminais. O objetivo do jogo é que o usuário descubra uma palavra secreta de cinco letras em um número limitado de tentativas, com feedback visual por cores para auxiliar o jogador. O projeto destaca técnicas de manipulação de arquivos, controle de fluxo, tratamento de entrada do usuário e interface via terminal com códigos ANSI para colorização.

**Palavras-chave:** Jogos em terminal, Linguagem C, Wordle-like, ANSI escape codes, Persistência em JSON

# Sumario

Resumo .....	3
1. Introdução .....	5
2. Objetivos .....	5
2.1 Objetivo Geral.....	5
2.2 Objetivos Específicos .....	5
3. Metodologia .....	6
3.1 Linguagem e Ferramentas .....	6
3.2 Estrutura do Código .....	6
3.3 Fluxo do Jogo .....	7
4. Implementação .....	7
4.1 Estruturas de Dados.....	7
4.2 Funcionalidades Principais.....	7
4.2.1 Sistema de Dicas .....	7
4.2.2 Feedback Visual .....	8
4.2.3 Teclado Virtual.....	8
4.2.4 Persistência de Dados.....	8
4.3 Tratamento de Input.....	8
5. Resultados.....	8
6. Discussão.....	9
7. Conclusão .....	9
8. Postagens.....	9
Referências.....	10
Anexos .....	11

# 1. Introdução

O **CodleC** é um jogo de adivinhação de palavras desenvolvido em **linguagem C**, inspirado no *Wordle* (The New York Times, 2025) e em jogos similares como *Termooo* e *Letreco*. O projeto combina conceitos de **programação estruturada**, **manipulação de terminal** e **persistência de dados** para criar uma experiência interativa em modo texto, com feedback visual baseado em cores ANSI (ANSI Escape Codes, 2025).

A motivação para o desenvolvimento do CodleC surgiu da necessidade de explorar técnicas de programação em C, como:

- Entrada/saída não bloqueante (uso de `termios` no Unix e `_getch()` no Windows).
- Gerenciamento dinâmico de palavras via arquivos `.txt`.
- Feedback visual acessível (cores para daltonismo).

O jogo foi testado em ambientes **Linux/Unix e Windows**, garantindo portabilidade. Além disso, vídeos de referência (YouTube, 2025) foram utilizados para validar a jogabilidade e a usabilidade.

## 2. Objetivos

### 2.1 Objetivo Geral

Desenvolver um jogo de palavras interativo em C, com interface colorida no terminal, múltiplos níveis de dificuldade e persistência de resultados.

### 2.2 Objetivos Específicos

- Implementar um sistema de feedback visual baseado em cores (ANSI escape codes).
- Oferecer diferentes níveis de dificuldade (Fácil, Médio, Difícil e Demo).
- Criar um sistema de dicas limitadas com cooldown.
- Salvar resultados em um arquivo JSON para histórico.
- Garantir compatibilidade multiplataforma (Windows e Unix/Linux).

## 3. Metodologia

O CodleC foi desenvolvido utilizando a linguagem C padrão, com foco em portabilidade para sistemas operacionais Linux, macOS e Windows (via terminal compatível). A estrutura do projeto é composta pelos seguintes módulos principais:

- **Carregamento de palavras:** Leitura de arquivos de texto (palavras.txt e palavras\_dificéis.txt) contendo listas de palavras válidas para cada nível de dificuldade.
- **Validação de entrada:** Funções para assegurar que a entrada do usuário consiste em palavras válidas, compostas apenas por caracteres alfabéticos e com tamanho correto.
- **Lógica de jogo:** Implementação da mecânica de comparação entre a tentativa do jogador e a palavra secreta, gerando um resultado que indica letras corretas, letras na posição errada e letras incorretas.
- **Interface colorida:** Utilização de códigos ANSI para imprimir letras em cores diferentes no terminal, melhorando a experiência visual do jogador.
- **Persistência de resultados:** Registro dos resultados em formato JSON para possibilitar acompanhamento do desempenho em múltiplas partidas.
- **Controles adicionais:** Funções para pausar e reiniciar o jogo durante a execução.

### 3.1 Linguagem e Ferramentas

- **Linguagem:** C (padrão C99).
- **Compiladores:** GCC (Linux/macOS) ou MinGW (Windows).
- **Terminal:** Suporte a códigos ANSI para cores.

### 3.2 Estrutura do Código

O projeto segue uma abordagem modular, com funções organizadas em:

- **Inicialização do jogo** (init\_game).
- **Processamento de palpites** (process\_guess, calculate\_feedback).
- **Interface do usuário** (display\_game\_board, display\_keyboard).

- **Controle de dicas** (use\_hint, can\_use\_hint).
- **Persistência de dados** (salvar\_resultado\_json).

### 3.3 Fluxo do Jogo




#### 1. Menu Principal:

- a. Opções: Jogar, Como Jogar, Resultados, Sair.

#### 2. Seleção de Dificuldade:

- a. Fácil (7 tentativas), Médio (6), Difícil (5), Demo (palavra fixa "TESTE").

#### 3. Jogabilidade:

- a. O jogador insere palavras e recebe feedback visual (  ,  ,  ).
- b. Pode usar dicas (H) ou pausar (P).

#### 4. Fim de Jogo:

- a. Vitória: Salva resultado em resultados.json.
- b. Derrota: Revela a palavra secreta.

## 4. Implementação

### 4.1 Estruturas de Dados

- **GameState:** Armazena o estado atual do jogo (palavra-alvo, palpites, feedback, dicas usadas, etc.).
- **Listas de Palavras:**
  - word\_list: Palavras comuns (Fácil/Médio).
  - hard\_word\_list: Palavras complexas (Difícil).

### 4.2 Funcionalidades Principais

#### 4.2.1 Sistema de Dicas

- Máximo de **4 dicas por partida**.
- **Cooldown de 30 segundos** entre dicas.

- Revela uma letra aleatória ainda não descoberta.

#### 4.2.2 Feedback Visual

- **Verde** (■): Letra correta na posição correta.
- **Amarelo** (■): Letra existe, mas em posição errada.
- **Cinza** (□): Letra não está na palavra.

#### 4.2.3 Teclado Virtual

Mostra o status de cada letra já testada.

#### 4.2.4 Persistência de Dados

Salva resultados em resultados.json no formato:

```
{"palavra": "TESTE", "tentativas": 3, "dificuldade": "FÁCIL"}
```

### 4.3 Tratamento de Input

#### **Multiplataforma:**

- Windows: `_getch()` (entrada sem buffer).
- Unix: `termios` (modo raw para captura imediata).

## 5. Resultados

O jogo permite ao usuário escolher entre três níveis de dificuldade, ajustando o número máximo de tentativas e o conjunto de palavras a serem escolhidas. A interface gráfica de terminal apresenta as letras em verde, amarelo ou cinza, de acordo com a correspondência com a palavra secreta, facilitando a dedução correta.

Além disso, o sistema armazena um histórico de partidas em arquivo resultados.json, que pode ser consultado pelo usuário a qualquer momento. O jogo foi testado em ambientes Linux e Windows, mostrando compatibilidade e funcionamento consistente.



## 6. Discussão

O CodleC demonstra que é possível criar jogos interativos com recursos visuais atrativos mesmo em ambientes restritos como o terminal, utilizando recursos simples da linguagem C e manipulação de caracteres ASCII. A escolha da cor para feedback aumenta a usabilidade sem a necessidade de interfaces gráficas complexas.

Como limitações, o programa depende de listas de palavras externas, que devem ser gerenciadas separadamente. Futuras melhorias podem incluir expansão do dicionário, suporte a outras línguas, e implementação de modos multiplayer.

## 7. Conclusão

O CodleC representa uma ferramenta didática eficiente para aprender e praticar programação em C, abordando conceitos de manipulação de arquivos, tratamento de strings, controle de fluxo, e interface baseada em terminal. O projeto serve tanto para entretenimento quanto para desenvolvimento de habilidades técnicas, destacando-se pela simplicidade e portabilidade.

## 8. Postagens

### Links de Publicação

- [GitHub](#)
- [Medium](#)
- [Dev.io](#)

## Referências

ANSI escape codes. Disponível em:

[https://en.wikipedia.org/wiki/ANSI\\_escape\\_code](https://en.wikipedia.org/wiki/ANSI_escape_code). Acesso em: 3 jun. 2025.

TERMOOOO. Disponível em: <https://term.ooo>. Acesso em: 30 mai. 2025.

Letreco. Disponível em: <https://letreco.org>. Acesso em: 30 mai. 2025.

Wordle game. Disponível em: <https://www.nytimes.com/games/wordle/index.html>.

Acesso em: 29 mai. 2025.

YouTube. **Text-Based Wordle in Python under 15 Minutes!** Disponível em:

<https://www.youtube.com/watch?v=J6h7D2iQmBU>. Acesso em: 2 jun. 2025.

YouTube. **Text-Based Wordle in C under 25 Minutes! (Part 1 of 2)**. Disponível em:

<https://www.youtube.com/watch?v=oO77gXCWCMo>. Acesso em: 2 jun. 2025.

YouTube. **Text-Based Wordle in C under 25 Minutes! (Part 2 of 2)**. Disponível em:

<https://www.youtube.com/watch?v=J6h7D2iQmBU>. Acesso em: 2 jun. 2025.

# Anexos

**Código Fonte :** Main.c

**Executáveis :** Main.exe **Links de Publicação**