Gabriel Tenório
& Vinicius Arima

RELATÓRIO TÉCNICO

Professor: João Victor Tinoco
Programação Imperativa e Funcional

1. Introdução

Em um mar de bits e bytes, fomos convocados para uma missão crítica: recriar o lendário conflito naval utilizando apenas a essência pura da linguagem C. Missão dada, missão cumprida.

2. Ambiente e Restrições

Todo o desenvolvimento foi realizado exclusivamente na linguagem C, seguindo o padrão C99. O projeto absteve-se do uso de bibliotecas externas ou interfaces gráficas complexas, focando na performance e na portabilidade.

3. Regras e Mecânica do Jogo

A implementação segue fielmente as regras clássicas do jogo. O tabuleiro foi projetado para ser dinâmico, permitindo configurações que variam do padrão 10x10 até dimensões de 26x26.

A frota é composta pelos tradicionais Porta-aviões, Encouraçados, Cruzadores e Destroyers. Para o posicionamento, oferecemos ao jogador a liberdade de escolha entre um modo manual, onde coordena cada navio estrategicamente, e um modo automático, impulsionado por um algoritmo inteligente que distribui a frota aleatoriamente sem sobreposições. O combate ocorre em turnos alternados, oferecendo feedback imediato sobre cada disparo e detectando automaticamente a vitória quando uma frota é aniquilada.

4. Arquitetura do Projeto (Modularização)

O sistema foi arquitetado em módulos com responsabilidades isoladas. O arquivo `main.c` atua como o orquestrador central, gerenciando o fluxo entre o menu e o jogo. Os módulos `board.c` e `fleet.c` encarregam-se, respectivamente, da alocação da matriz do tabuleiro e do gerenciamento da frota. A lógica de regras, validação de tiros e controle de fluxo reside em `game.c`, enquanto toda a interação com o usuário foi isolada em `io.c`, mantendo o núcleo do jogo agnóstico à interface. Por fim, `rnd.c` provê os utilitários necessários para a geração aleatória de jogadas do oponente.

5. Estruturas de Dados e Fluxo

A espinha dorsal do projeto baseia-se no uso avançado de estruturas (`structs`) e ponteiros. A estrutura `Board` foi desenhada para armazenar as dimensões e um ponteiro para as células, permitindo a criação de tabuleiros de qualquer tamanho em tempo de execução.

Já a estrutura `Player` encapsula todo o estado do jogador, incluindo tabuleiro, frota e estatísticas, o que facilita a manipulação do jogo através de referências de memória. O fluxo de execução é linear e intuitivo, iniciando-se com um menu interativo, passando pela configuração da partida e culminando no loop principal de combate, sempre com rigoroso controle de alocação e liberação de recursos.

6. Requisitos Técnicos Obrigatórios

O projeto demonstra pleno domínio dos conceitos fundamentais exigidos pela disciplina. Destaca-se o uso intensivo de alocação dinâmica via `malloc` para a criação dos tabuleiros e vetores de navios, garantindo flexibilidade total.

Além disso, houve um cuidado extremo com a gestão de memória, implementando a liberação explícita de todos os recursos alocados através de funções de limpeza ao final da execução, prevenindo vazamentos de memória. A passagem de dados por referência via ponteiros foi adotada em todas as funções críticas para otimizar o desempenho.

7. Automação e Entrega

Para elevar o nível de profissionalismo da entrega, incluímos um sistema de build automatizado via Makefile. Com um único comando, o sistema compila todos os módulos e executa o jogo, assegurando compatibilidade e facilidade de teste tanto em ambientes Windows quanto em sistemas baseados em Unix, como Linux e macOS.

8. Conclusão

Este projeto consolidou nosso entendimento sobre a programação estruturada em C de forma prática e desafiadora. Mais do que apenas codificar as regras de um jogo, enfrentamos e superamos os obstáculos de gerenciar manualmente a memória e estruturar uma aplicação complexa e escalável. O resultado final é um software eficiente, organizado e pronto para a batalha, refletindo nossa evolução técnica ao longo da disciplina.

(Veja, na próxima página, amostras do código)

AMOSTRAS DO NOSSO CÓDIGO:

Aqui você pode ver ele funcionando ao dar o comando “make”

```
PS C:\Users\gabri\Downloads\pif-naval-c-main\pif-naval-c-main> make
gcc -o jogo.exe src/*.c -W -Wall -ansi -pedantic -std=c99
./jogo.exe

|============== BATALHA NAVAL ============== |
1) Novo Jogo
2) Configs
3) Sair
Manda a boa:
```

```
--> VEZ DE gab (JOGADOR 1) <--

Aperta ENTER pra ver o jogo e mandar bala...

--- AREA INIMIGA ---
  A B C D E F G H I J
 1 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 2 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 3 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 4 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 5 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 6 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 7 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 8 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 9 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
10 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~

--- TUA FROTA ---
  A B C D E F G H I J
 1 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 2 ~ ~ ~ ~ ~ ~ ~ ~ N ~
 3 ~ ~ ~ N N N N N N ~
 4 ~ ~ N ~ ~ ~ ~ ~ ~ ~
 5 ~ ~ N ~ ~ ~ ~ ~ ~ ~
 6 ~ ~ ~ ~ ~ N N N N ~
 7 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 8 N N N ~ ~ ~ ~ ~ ~ ~
 9 ~ ~ ~ ~ N N N ~ ~ ~
10 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~

gab, sua vez de atirar. Digite Coordenada ex: A5
```

```
gab, sua vez de atirar. Digite Coordenada ex: A5 a5
Agua!
```

```
vin, sua vez de atirar. Digite Coordenada ex: A5 3d
Coordenada errada, tenta de novo bro.
Digite Coordenada ex: A5 d3
ACERTOU! Fogo a bordo!

=============== TURNO 2 ===============

--> VEZ DE gab (JOGADOR 1) <--

Aperta ENTER pra ver o jogo e mandar bala...
```

O QUE ACONTECE QUANDO ALGUÉM VENCE?

```
VITORIA! vin AMASSOU EM 19 TURNOS!

--- Stats do vin ---
Tiros: 19
Acertos: 19
Precisao: 100.00%

Como ficou o tabuleiro dele:
   A B C D E F G H I J
 1 * ~ ~ ~ ~ * * ~ * ~
 2 ~ X N N ~ ~ ~ ~ * ~
 3 * ~ ~ ~ ~ * ~ ~ ~ ~
 4 * ~ ~ ~ N N N N * ~
 5 * ~ ~ ~ N N N ~ ~ ~
 6 * ~ ~ ~ ~ ~ ~ ~ N ~
 7 * ~ ~ ~ ~ ~ ~ ~ N N
 8 * ~ ~ ~ ~ ~ ~ ~ N N
 9 * ~ ~ ~ N ~ ~ ~ X ~
10 * ~ ~ ~ N ~ ~ ~ X *

--- Stats do gab ---
Tiros: 19
Acertos: 3
Precisao: 15.79%

Como ficou o tabuleiro dele:
   A B C D E F G H I J
 1 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 2 ~ ~ ~ ~ ~ ~ ~ ~ X ~
 3 ~ ~ ~ X X X X X X ~
 4 ~ ~ X ~ ~ ~ ~ ~ ~ ~
 5 ~ ~ X ~ ~ ~ ~ ~ ~ ~
 6 ~ ~ ~ ~ ~ X X X X ~
 7 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
 8 X X X ~ ~ ~ ~ ~ ~ ~
 9 ~ ~ ~ ~ X X X ~ ~ ~
10 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
```