PIF – SI 2025.2
Jogo da Batalha Naval

Curso de Sistemas de Informação – CESAR School

Professor: João Victor Tinoco

1. Objetivo

O objetivo deste projeto é implementar, em grupos de três alunos, o clássico jogo **Batalha Naval** utilizando exclusivamente as bibliotecas básicas da linguagem C. O projeto deve demonstrar o domínio de conceitos de **Structs**, **Ponteiros**, **Alocação Dinâmica de Memória (malloc e realloc)** e os **fluxos básicos de controle**.

2. Restrições e Ambiente

- Linguagem: C
- Bibliotecas permitidas: `stdio.h`, `stdlib.h`, `string.h`, `time.h`, `ctype.h`, `stdbool.h`.
- É proibido o uso de bibliotecas externas ou recursos gráficos.

3. Regras do Jogo

- Tabuleiro padrão de 10x10 posições (configurável entre 6 e 26).
- Frota mínima:
  - 1 Porta-aviões (5 células)
  - 1 Encouraçado (4 células)
  - 2 Cruzadores (3 células)
  - 2 Destroyers (2 células)
- Cada jogador posiciona seus navios manualmente ou de forma automática (aleatória).
- Os jogadores se alternam realizando disparos, informando coordenadas como "B5".
- O jogo termina quando todos os navios de um jogador forem afundados.

4. Estrutura do Projeto

4.1 Módulos sugeridos

```
/src
  board.h / board.c   -> Estrutura do tabuleiro
  fleet.h / fleet.c   -> Definição e controle dos navios
  game.h  / game.c    -> Regras do jogo e controle de turnos
  io.h    / io.c      -> Entrada e saída de dados (CLI)
  rnd.h   / rnd.c     -> Geração de números aleatórios
  main.c               -> Ponto de entrada do programa
```

4.2 Estruturas principais

```
typedef enum { CELL_WATER, CELL_SHIP, CELL_HIT, CELL_MISS } CellState;

typedef struct {
    CellState state;
    int ship_id; // -1 se não houver navio
} Cell;

typedef struct {
    int rows, cols;
    Cell *cells; // malloc(rows * cols)
} Board;

typedef enum { ORIENT_H, ORIENT_V } Orientation;

typedef struct {
    char name[20];
    int length;
    int hits;
    int placed;
} Ship;

typedef struct {
    Ship *ships;
    int count;
} Fleet;

typedef struct {
    Board board;
    Board shots;
    Fleet fleet;
    char nickname[32];
} Player;

typedef struct {
    Player p1, p2;
    int current_player;
    int game_over;
} Game;
```

5. Fluxo Básico do Programa

1. Menu inicial: Novo jogo | Configurações | Sair.
2. Configurações: tamanho do tabuleiro e modo de posicionamento.
3. Posicionamento dos navios: manual ou automático.

4. Loop de turnos: jogadores alternam tiros até a vitória.

5. Exibição do vencedor e estatísticas.

6. Exemplo de Execução (CLI)

```
=== BATALHA NAVAL ===
1) Novo jogo
2) Configurações
3) Sair
> 1

Informe apelido do Jogador 1: Alice
Informe apelido do Jogador 2: Bob

Tamanho do tabuleiro (6-26):
> 10
Posicionamento (M)anual ou (A)utomático?
> A

[Frota de Alice e Bob posicionadas automaticamente]

-- Turno de Alice --
Digite coordenada do tiro (ex.: E5):
> E5
Resultado: ÁGUA.

-- Turno de Bob --
> B2
Resultado: ACERTOU no Cruzador!

-- Turno de Alice --
> B3
Resultado: AFUNDOU Cruzador!

*** FIM DE JOGO ***
Vencedor: Alice
Tiros: 31 | Acertos: 17 | Precisão: 54.8%

--- Estado final dos tabuleiros ---

Tabuleiro de Alice (seus navios):
  A B C D E F G H I J
1 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
2 ~ S S S ~ ~ ~ ~ ~ ~
3 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
4 ~ ~ X X ~ ~ ~ ~ ~ ~
5 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
```

```
6 ~ ~ ~ ~ S S ~ ~ ~ ~
7 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
8 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
9 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
10~ ~ ~ ~ ~ ~ ~ ~ ~ ~

Legenda: S = Navio | X = Acertado | ~ = Água

Tabuleiro de Bob (visão real):
  A B C D E F G H I J
1 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
2 X X X ~ ~ ~ ~ ~ ~ ~
3 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
4 ~ ~ X X X ~ ~ ~ ~ ~
5 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
6 ~ ~ ~ ~ S S S ~ ~ ~
7 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
8 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
9 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
10~ ~ ~ ~ ~ ~ ~ ~ ~ ~

Legenda: X = Acertado | S = Navio | ~ = Água

Mapa de tiros de Alice em Bob:
  A B C D E F G H I J
1 . . . . . . . . . .
2 . X X X . . . . . .
3 . . . . . . . . . .
4 . . X X X . . . . .
5 . . . . X . . . . .
6 . . . . . . . . . .
7 . . . . . . . . . .
8 . . . . . . . . . .
9 . . . . . . . . . .
10. . . . . . . . . .

Legenda: X = tiro acertado | . = tiro errado
```

7. Requisitos Obrigatórios

- Uso de `structs` para modelar entidades (tabuleiro, navio, jogador).
- Uso explícito de `malloc`, `realloc` e `free`.
- Passagem por ponteiros e manipulação via endereços.
- Validação rigorosa de entrada e limites de tabuleiro.
- Modularização adequada: separação de responsabilidades.

8. Critérios de Avaliação (100 pontos)

Critério	Pontuação
Design e Organização (módulos, nomes claros)	20 pts
Uso de structs, ponteiros, malloc/realloc, enums e fluxos de controle	35 pts
Fluxos e Regras (loop, tiros, vitória, validações)	25 pts
Qualidade de Código (sem warnings, legível, comentado)	20 pts

Penalizações automáticas:

- Acesso fora dos limites do tabuleiro: -10 pts.
- Jogo injogável (falhas graves de lógica): -30 pts.

9. Entregáveis

- Código-fonte completo (`.c`/`.h`) e Makefile.
- Arquivo `README.md` com instruções e decisões de design.
- Relatório técnico (este documento em PDF).

10. Dicas de Boas Práticas

- Prefira funções curtas (máx. 25 linhas) e coesas.
- Use `enum` em vez de números mágicos.
- Centralize entrada e saída em `io.c`.
- Libere memória na ordem inversa à alocação.

12. Conclusão

O projeto visa consolidar os fundamentos da programação estruturada em C, reforçando a importância da modularização, do controle de memória e do design limpo. O jogo da Batalha Naval é um excelente estudo de caso por envolver estruturas hierárquicas, manipulação de ponteiros e lógica de fluxo interativa.

"Programar bem é como jogar xadrez: toda jogada deve ter um propósito."