Trabalho de P3: Agente Interativo de Batalha Naval (Protocolo Textual)

---

1. Objetivos

Desenvolver um agente computacional autónomo, em **OCaml** ou **Prolog**, capaz de jogar Batalha Naval. O agente deve gerir o seu próprio tabuleiro, implementar uma estratégia de ataque inteligente e comunicar com um adversário (ou um script de teste) através de um protocolo de texto (`stdin`/`stdout`).

O agente não deve revelar nada acerca da linguagem em que foi implementado (nada de mensagens a anunciar a versão do Prolog ou do OCaml).

O projeto foca-se na gestão de estado e na implementação de uma estratégia que reage a feedback detalhado do adversário.

---

2. Descrição e Regras do Jogo

O jogo decorre num tabuleiro de **NxN** posições, em que N é um parâmetro cujo valor por omissão é 8. Cada "cano" mencionado refere-se a uma única célula (casa) do tabuleiro.

A «Frota»

Cada jogador dispõe da seguinte frota (total de 8 barcos):

- **1x Porta-aviões:** 5 células, em forma de "T".
- **1x Destroyer:** 4 células (linear).
- **2x Fragatas:** 3 células cada (linear).
- **3x Torpedeiros:** 2 células cada (linear).
- **1x Submarino:** 1 célula.

Formato dos Barcos

- **Barcos Lineares:** Ocupam N células consecutivas (horizontal ou verticalmente).
- **Porta-aviões (T):** Ocupa 5 células. É formado por uma linha de 3 células (ex: `(L, C-1)`, `(L, C)`, `(L, C+1)`) e um braço de 2 células perpendicular, ligado à célula central e estendendo-se na mesma direção (ex: `(L-1, C)`, `(L-2, C)`). O barco pode ser rodado em incrementos de 90°.

Regra de Posicionamento

O posicionamento inicial dos barcos deve obedecer a uma regra estrita: **os barcos não se podem tocar**, nem sequer nos cantos. Deve haver pelo menos uma célula (de água) a separar dois barcos em todas as direções (horizontal, vertical e diagonal).

---

3. Requisitos Funcionais e Protocolo

O agente deve funcionar num ciclo de turnos, lendo do stdin e escrevendo no stdout.

Gestão do Estado

O agente deve manter, no mínimo, dois tabuleiros:

- **Tabuleiro de Defesa:** Onde os seus barcos estão posicionados. Este tabuleiro é usado para responder aos ataques do adversário.
- **Tabuleiro de Ataque:** Onde o agente regista os seus próprios disparos e as respostas do adversário (água, tiro, afundado).

Protocolo de Comunicação

A comunicação é baseada em texto simples. Assume-se um tabuleiro 8x8 inicialmente vazio.

A primeira fase do jogo é a configuração, e é composta por comandos, a ler do stdin:

1. **opcionalmente**, um comando que redimensiona o tabuleiro

init N	Se omitido assume-se N=8. É obrigatório que os tabuleiros de ambos os agentes tenham a mesma dimensão.

2. uma fase de colocação de peças no tabuleiro, que pode ser uma **repetição de linhas** como esta:

barco NOME L1 C1 {L2 C2 ...}	Coloca um barco chamado **NOME** nas posições (**L1**,**C1**), (**L2**,**C2**), etc. Entende-se que este conjunto designa um único barco. **As linhas e colunas Li e Ci deverão estar no intervalo 0..N-1**

3. **ou** um comando que coloca "aleatóriamente" (pode ter estratégia) todos os barcos:

random	Inicializa o tabuleiro com os barcos pré-definidos (a "frota")

4. finalmente, um comando para dar início ao jogo, que pode ser um dos seguintes:

vou eu	Inicia a fase de jogo, este agente começa por dar um tiro no outro agente
vai tu	Inicia a fase de jogo, este agente começa por ler o tiro do outro agente

A segunda fase do jogo, são fases alternadas de ataque e resposta.

- Quando é a **vez do agente atacar**:

  O agente decide a sua jogada (L, C) e **envia** para o stdout:

tiro L C	Dispara um tiro para (L,C) e espera pela resposta do outro agente

  Depois de atacar, o agente deve ler uma linha do stdin com a resposta do agente que acabou de ser atacado, e que deverá ter uma das seguintes formas:

  1. água
     - O agente marca (L, C) como "água" no seu tabuleiro de ataque.
  2. tiro <barco> (ex: tiro fragata)
     - O agente marca (L, C) como "acerto" no seu tabuleiro de ataque.
     - O agente deve registar que acertou num <barco>, mas que este ainda não foi afundado.
  3. afundado <barco> (ex: afundado torpedeiro)
     - O agente marca (L, C) como "acerto".
     - O agente deve registar que o <barco> em (L, C) (e possivelmente noutras células, já "atiradas") está agora afundado.
  4. perdi
     - O agente regista que o outro agente perdeu, i.e. todos os seus navios foram afundados. Não haverá mais interação.

- Quando é a **vez do agente ser atacado** (receber um tiro):

  O agente deve **ler** o comando do adversário do stdin:

  ```
  tiro L C
  ```

  Seguidamente deve enviar a resposta: o agente atua em consequência do tiro no seu tabuleiro de defesa e responde imediatamente no stdout:

  1. Se (L, C) for água:

     ```
     água
     ```

  2. Se (L, C) acertar num barco, mas não for o último "cano" desse barco:

tiro <barco>
(Ex: **tiro destroyer**)

3. Se (L, C) acertar num barco, se for o último "cano" necessário para o afundar e que este for o último barco ainda não afundado:

   perdi

4. Se (L, C) acertar num barco e for o último "cano" necessário para o afundar:

   afundado <barco>
   (Ex: **afundado submarino** ou **afundado porta-aviões**)

Nota Importante: Para responder tiro ou afundado, o agente **tem** de gerir o estado de vida (células por acertar) de cada um dos seus barcos, individualmente.

---

4. Estratégia do Agente ("IA")

O agente deve implementar uma estratégia de ataque eficaz. Disparar aleatoriamente não é suficiente.

- **Requisito Mínimo (Modo "Caça"):** Após obter uma resposta tiro <barco>, o agente deve focar os seus próximos disparos nas células adjacentes (não-diagonais) à célula atingida, para tentar encontrar o resto do barco.
- **Requisito Essencial (Modo "Destruição"):** Se o agente obtiver dois acertos num barco (tiro <barco> duas vezes) que não seja o submarino ou o porta-aviões, deve ser capaz de inferir a sua orientação (horizontal ou vertical) e continuar a disparar nessa linha até receber a resposta afundado <barco>.
- **Uso da Informação afundado:** A resposta afundado é determinante. O agente deve usá-la para:
  1. Parar de procurar o resto desse barco.
  2. (Opcional, mas recomendado) Marcar todas as células adjacentes ao barco afundado como água (devido à regra de posicionamento), otimizando disparos futuros.

---

5. Entregáveis

- **Código fonte** completo e comentado (ficheiros .ml/.mli ou .pl).
- Um ficheiro README.md que descreva:
  - As estruturas de dados usadas para representar os tabuleiros e o estado do jogo.
  - Como compilar e executar o agente.

- Uma descrição clara da **estratégia de "IA"** implementada (como funciona o modo "Caça" e o modo "Destruição").

---

6. Outros

Os agentes deverão poder jogar entre si. Será organizado um torneio.

Os 3 primeiros grupos classificados terão uma bonificação respetiva de 2 pontos na nota do trabalho (1ª posição), 1 ponto (2ª posição) e 0.5 pontos (3ª posição).

---

7. Critérios de Avaliação

A avaliação do projeto focará três áreas principais:

- **Correção Funcional e Protocolo**
- **Estratégia do Agente ("IA")**
- **Qualidade e Clareza do Código**