UNIVERSIDADE LUSÓFONA

LP1 2024/2025 – Projeto 2025 Recurso - Radar Náutico

Pedro Arroz Serra, Daniel Silveira, Giosuè Muratore, Martijn Kuipers

Deadline: 28/06/2025 @ 8h59hrs no deisi-moodle *21-06-2025 v1.0*

1. Projeto Radar 2025 - Recurso

1.1 Objetivo

Desenvolver uma aplicação em **C** que simule o movimento de barcos, com atualização contínua no tempo. O projeto incide nos seguintes tópicos:

- Este projeto tem ser desenvolvido na linguagem de programação C.
- Ponteiros e gestão dinâmica de memória
- Listas ligadas
- Estruturas compostas
- Leitura e escrita de ficheiros
- Organização modular do código

1.2 Requisitos Técnicos

O não cumprimento destes requisitos implica uma penalização de 100%

- Utilização obrigatória de listas ligadas.
- É proibido utilizar arrays estáticos para armazenar objetos de natureza dinâmica.
- Não é permitida a utilização de variáveis globais ou estáticas
- O programa não deve ter memory leaks.
- Utilização obrigatória das funções `malloc` e `free`.
- O código deve compilar sem erros ou avisos, utilizando o `gcc` com as seguintes flags: `-Wall -Wextra -g -Wvla -Wpedantic -Wdeclaration-after-statement -lm`
- Implementação de todos parâmetros de entrada
- O trabalho é individual e deve ser desenvolvido e submetido como tal.

1.3 Formato do Ficheiro de Entrada e Saída

Formato de cada linha:

```
<ID> <LATITUDE> <LONGITUDE> <ÂNGULO> <VELOCIDADE> <TIPO>
```

Exemplo:

```
A 11 20 0 1 0
B 15 25 270 3 0
C 39 20 180 1 0
D 30 25 270 3 0
```

Nota: Os ângulos possíveis são múltiplos de 45 graus, ou seja, 0, 45, 90, 135, 180, 225, 270 e 315 graus. Daí o aluno deve converter os ângulos e velocidade para um vetor de movimento (dx, dy) que representa a direção e velocidade do barco horizontalmente e verticalmente, seguindo os ponteiros do relógio.

Por exemplo:

- A primeira linha é o Barco A, na posição X=10, Y=20, com um ângulo de 0 graus, logo anda para a direita com uma velocidade no sentido dy de 1 casa por segundo.
- O barco B está na posição X=15, Y=25, com um ângulo de 270 graus (para cima) e uma velocidade vy de −3 casas por segundo.
- O barco C está na posição X=40, Y=20, com um ângulo de 180 graus (para a esquerda) e uma velocidade vx de −1 casa por segundo.

A última coluna é o tipo de barco, que é um número entre 1 e 13, detalhado na próxima secção.

1.4 Tipos de Barco

Cada barco possui um tipo funcional que define o seu comportamento na simulação.

Tipo	Nome	Comportamento/Funcionalidade Especial
0	Normal	Uma embarcação normal
1	Drone	Anda em zig zag, a velocidade horizontal é 0 nos frames impares, e velocidade vertical é 0 nos frames pares e inverte a velocidade horizontal ou vertical quando chega ao fronteira da grelha
2	Submarino	Aparece e desaparece da grelha de 5 em 5 frames
3	Veleiro	Move ao dobro da velocidade horizontal inicial se o barco esta a ir para direita (aumentar X). Se a próxima posição do Veleiro o colocar fora dos limites esquerdo ou direito da grelha, o barco não deve ser removido. Em vez disso, ele deve executar uma viragem de 180 graus. Quando faz a viragem mantem a posição.

1.5 Menu da Simulação

Se o parâmetro da linha de comando contém frames diferente de 0, o programa deve automaticamente avançar a simulação X frames.

A opção 1 - Atualizar simulação, deve avançar a simulação X frames.

- Pergunta quantos frames quer andar. O tempo é dado em segundos, e cada frame corresponde a 1 segundo.
- O programa deve avançar a simulação o número de frames escrito, atualizando a posição de cada barco.
- Se barcos colidirem, deve remover ambos do radar.
- Após atualizar, grava no ficheiro de saída o resultado da simulação. O nome do ficheiro de saída é o parâmetro da linha de comando.

A opção 2 - Adicionar um barco novo ou atualizar um barco.

- Deverá perguntar
  - o nome do barco
  - a posição inicial (latitude e longitude)
  - a direção (ângulo)
  - a velocidade.
- O barco é adicionado à lista de barcos.
- Se o barco já existir, deve ser alterado.

A opção 3 - listar barcos que irão colidir no futuro.

- Só considera colisão quando dois barcos estão na mesma posição.

- Só considera colisões que ocorrem dentro do radar.
- Apresenta coordenada onde vão ocorrer a colisões.

A opção 4 - andar com o tempo para trás X frames.
- Por exemplo, se temos um histórico com 10 segundos:
  - Deve pedir "Quantos frames deseja voltar?".
  - Se responder 3, a simumação executa como tivessemos no sétimo frame (10 - 3) e gravar no ficheiro.

A opção 5 - velocidade média de um barco
- Calcula o número de casas percorridas horizontalmente (dx) e verticalmente (dy) desde o frame 0 até ao frame atual.
- A distância percorrida é a fórmula "sqrt(dx **dx + dy** dy)", onde "sqrt" é a função raiz quadrada da biblioteca math.h.
- A velocidade média obtém-se da divisão da distância pelo número de frames passados desde o início do programa.

A opção 6 - Altera corrente
- O programa pede a novo valor da velocidade de corrente e novo angulo da corrente

Exemplo:

```
Nova velocidade da corrente: 2
Novo ângulo da corrente (multiplo de 45): 45
```

A opção 0 - termina o programa
- Guarda o estado atual da simulação no ficheiro de saída.
- Se fizemos o que foi dito na opção 4, deve guardar o frame 7 antes de terminar.
- De notar que o ficheiro de saída tem o mesmo formato que o ficheiro de entrada, mas exclui os barcos que afundaram ou saíram do radar.

```
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
```

1.6 Parâmetros de Entrada

Ao executar o programa, o utilizador deve fornecer os seguintes parâmetros:

- Ficheiro de entrada com a lista de barcos (ver secção "Ficheiro de Entrada")
- Dimensão da grelha, que são dois números (ex: 20x80)
- Angulo de corrente (multiplos de 45º)
- Velocidade de corrente
- Número de frames a simular (ex: 1000)
- Ficheiro de saída para guardar o estado da simulação

Exemplo de execução:

```
./radar antes.txt 20x80 45 2 1000 depois.txt
```

1.7 Formato do Ficheiro de Entrada e Saída

Formato de cada linha:

```
<ID> <LATITUDE> <LONGITUDE> <ÂNGULO> <VELOCIDADE> <TIPO>
```

Exemplo:

```
A 10 20 0 1 0
B 15 25 270 3 0
C 40 20 180 1 0
D 30 25 270 3 0
```

Nota: Os ângulos possíveis são múltiplos de 45 graus, ou seja, 0, 45, 90, 135, 180, 225, 270 e 315 graus. Daí o aluno deve converter os ângulos e velocidade para um vetor de movimento (dx, dy) que representa a direção e velocidade do barco horizontalmente e verticalmente, seguindo os ponteiros do relógio.

Por exemplo:

- A primeira linha é o Barco A, na posição X=10, Y=20, com um ângulo de 0 graus, logo anda para a direita com uma velocidade no sentido dy de 1 casa por segundo.
- O barco B está na posição X=15, Y=25, com um ângulo de 270 graus (para cima) e uma velocidade vy de −3 casas por segundo.
- O barco C está na posição X=40, Y=20, com um ângulo de 180 graus (para a esquerda) e uma velocidade vx de −1 casa por segundo.

A última coluna é o tipo de barco, que é um número entre 1 e 13, detalhado na próxima secção.

2. Exemplos

Considerar em todos os exemplos o ficheiro de entrada “antes.txt” dado no início do enunciado e gerar o ficheiro de saída “depois.txt”.

2.1 Exemplo de execução do programa, avança 1 frame manualmente via menu

```
./main antes.txt 80x80 0 0 0 depois.txt

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 1
Quantos frames deseja avancar? 1
Frame 1 guardado com sucesso em depois.txt
Simulacao atualizada para o frame 1

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 0
Frame 1 guardado com sucesso em depois.txt
A sair do programa...
```

Conteudo do ficheiro de saida “depois.txt” gerado pelo programa:

```
A 12 20 0 1 0
B 15 22 270 3 0
C 38 20 180 1 0
D 30 22 270 3 0
```

2.2 Exemplo de execução do programa 5 frames automatico

```
./main antes.txt 80x80 0 0 5 depois.txt
Frame 1 guardado com sucesso em depois.txt
Frame 2 guardado com sucesso em depois.txt
Frame 3 guardado com sucesso em depois.txt
Frame 4 guardado com sucesso em depois.txt
Frame 5 guardado com sucesso em depois.txt
Simulacao atualizada para o frame 5

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 0
Frame 5 guardado com sucesso em depois.txt
A sair do programa...
```

Conteudo do ficheiro de saida “depois.txt” gerado pelo programa:

```
A 16 20 0 1 0
B 15 10 270 3 0
C 34 20 180 1 0
D 30 10 270 3 0
```

2.3 Exemplo de alterar barco

```
 ./main antes.txt 80x80 0 0 0 depois.txt

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 2

=== Inserir/Alterar Barco ===
Nome do barco (uma letra): B
Posicao inicial (latitude longitude): 69 69
angulo (multiplo de 45): 180
Velocidade: 99
Tipo do barco (0-3): 2
Barco B alterado com sucesso.
Barco B adicionado com sucesso.

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 0
Frame 0 guardado com sucesso em depois.txt
A sair do programa...
```

Conteudo do ficheiro de saida “depois.txt” gerado pelo programa:

```
B 69 69 180 99 2
D 30 25 270 3 0
C 39 20 180 1 0
A 11 20 0 1 0
```

2.4 Exemplo de adicionar novo barco

Conteudo do ficheiro de saida “depois.txt” gerado pelo programa:

```
./main antes.txt 80x80 0 0 0 depois.txt

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
0. Sair
Escolha uma opcao: 2

=== Inserir/Alterar Barco ===
Nome do barco (uma letra): X
Posicao inicial (latitude longitude): 19 75
Angulo (múltiplo de 45): 45
Velocidade: 10
Tipo do barco (0-3): 3
Barco X adicionado com sucesso.

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
0. Sair
Escolha uma opcao: 0
Frame 0 guardado com sucesso em depois.txt
A sair do programa...
```

Conteudo do ficheiro de saida “depois.txt” gerado pelo programa:

```
X 19 75 45 10 3
D 30 25 270 3 0
C 39 20 180 1 0
B 15 25 270 3 0
A 11 20 0 1 0
```

2.5 Exemplo de colisão

```
./main antes.txt 80x80 0 0 0 depois.txt

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
0. Sair
Escolha uma opcao: 3

=== Previsão de Colisões ===
Colisao prevista entre barcos C e A:
  Posicao prevista da colisao: (25,20)

Total de colisões previstas: 1

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
0. Sair
Escolha uma opcao: 0
Frame 0 guardado com sucesso em depois.txt
A sair do programa...
```

2.6 Exemplo de rastrear, avançamos simulação 10 frames e recuamos 5 frames.

```
./main antes.txt 80x80 0 0 10 depois.txt
Frame 1 guardado com sucesso em depois.txt
Frame 2 guardado com sucesso em depois.txt
Frame 3 guardado com sucesso em depois.txt
Frame 4 guardado com sucesso em depois.txt
Frame 5 guardado com sucesso em depois.txt
Frame 6 guardado com sucesso em depois.txt
Frame 7 guardado com sucesso em depois.txt
Frame 8 guardado com sucesso em depois.txt
Frame 9 guardado com sucesso em depois.txt
Frame 10 guardado com sucesso em depois.txt
Simulacao atualizada para o frame 10

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 4

=== Rastrear Historico Reverso ===
Quantos frames deseja voltar? 10

Estado do frame 0:
Barco D: posicao (30,25), velocidade (0,-3)
Barco C: posicao (39,20), velocidade (-1,0)
Barco B: posicao (15,25), velocidade (0,-3)
Barco A: posicao (11,20), velocidade (1,0)

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 0
Frame 6 guardado com sucesso em depois.txt
A sair do programa...
```

Conteudo do ficheiro de saida “depois.txt” gerado pelo programa:

```
A 11 20 0 1 0
B 15 25 270 3 0
C 39 20 180 1 0
D 30 25 270 3 0
```

2.7 Exemplo de velocidade média após avançarmos a simulação 3 frames

```
./main antes.txt 80x80 0 0 3 depois.txt
Frame 1 guardado com sucesso em depois.txt
Frame 2 guardado com sucesso em depois.txt
Frame 3 guardado com sucesso em depois.txt
Simulacao atualizada para o frame 3

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 5

=== Velocidade Media de um Barco ===
Nome do barco (uma letra): A

Estatisticas do barco A:
  Posicao inicial: (11,20)
  Posicao atual: (14,20)
  Distancia percorrida: 3.00 casas
  Numero de frames: 3
  Velocidade media: 1.00 casas/frame

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 0
Frame 3 guardado com sucesso em depois.txt
A sair do programa...
```

2.8 Exemplo de alterar a corrente

```
 ./main antes.txt 80x80 0 0 3 depois.txt
Frame 1 guardado com sucesso em depois.txt
Frame 2 guardado com sucesso em depois.txt
Frame 3 guardado com sucesso em depois.txt
Simulacao atualizada para o frame 3

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 6

=== Alterar Corrente ===
Nova velocidade da corrente: 10
Novo ângulo da corrente (multiplo de 45): 45
Corrente alterada para velocidade 10 e ângulo 45.

=== MENU DA SIMULACAO ===
1. Atualizar simulacao
2. Inserir ou alterar barco
3. Previsao de colisoes
4. Rastrear historico reverso
5. Velocidade media de um barco
6. Altera corrente
0. Sair
Escolha uma opcao: 0
Frame 3 guardado com sucesso em depois.txt
A sair do programa...
```

2.9 Exemplos dos tipos de barco a implementar

2.9.1 Barco Normal

Cada barco move em cada sentido, são todos normais, ou seja, não têm comportamentos especiais. Atenção: Neste exemplo nao ha corrente (velocidade e angulo a zero).

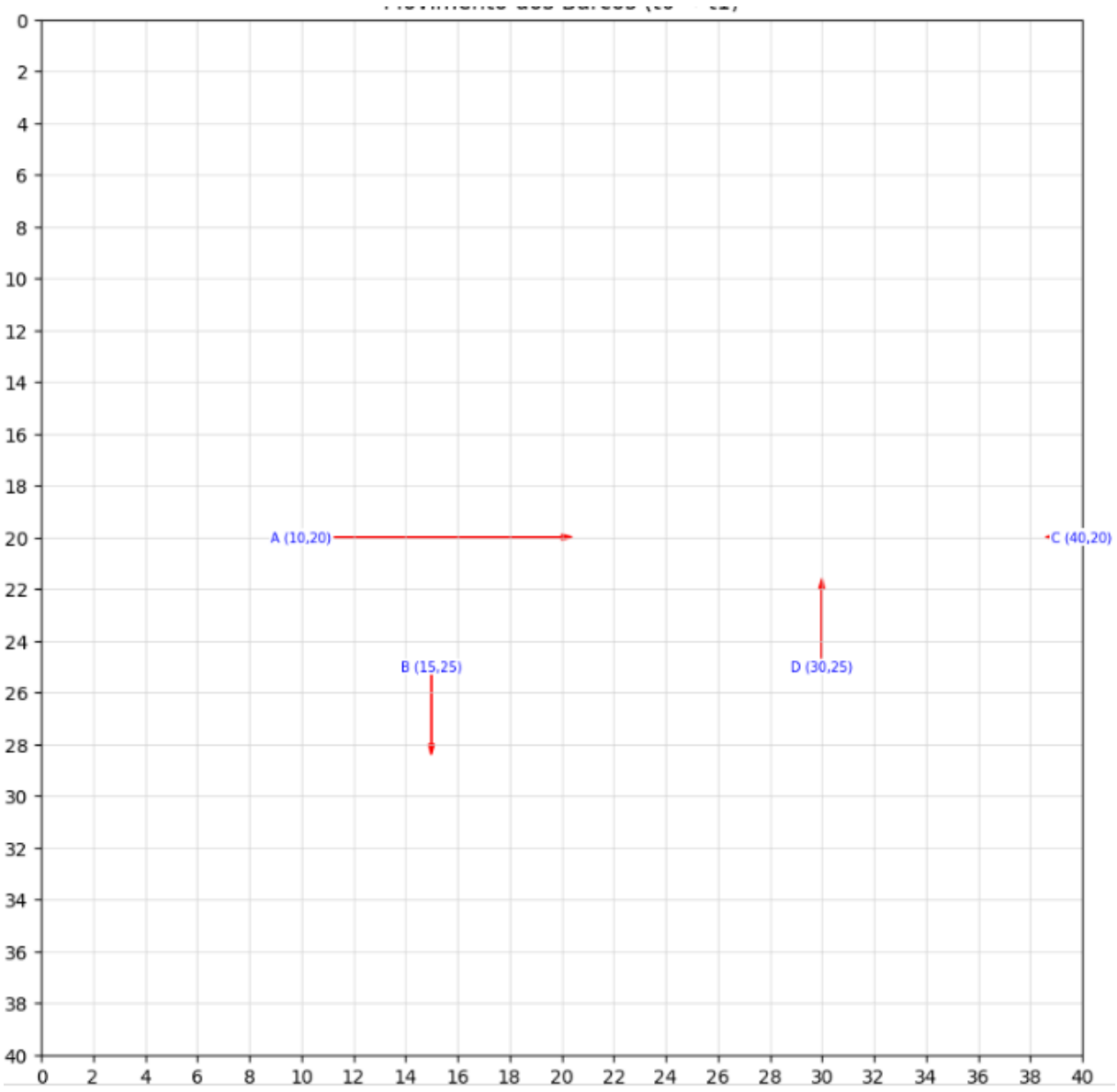
2.9.1.a Movimento do barco normal

Antes:

```
A 10 20 0 10 0
B 15 25 90 3 0
C 40 20 180 1 0
D 30 25 270 3 0
```

Depois de 1 segundo:

```
A 20 20 0 10 0
B 15 28 90 3 0
C 39 20 180 1 0
D 30 22 270 3 0
```



2.9.1.b Movimento do barco normal com corrente

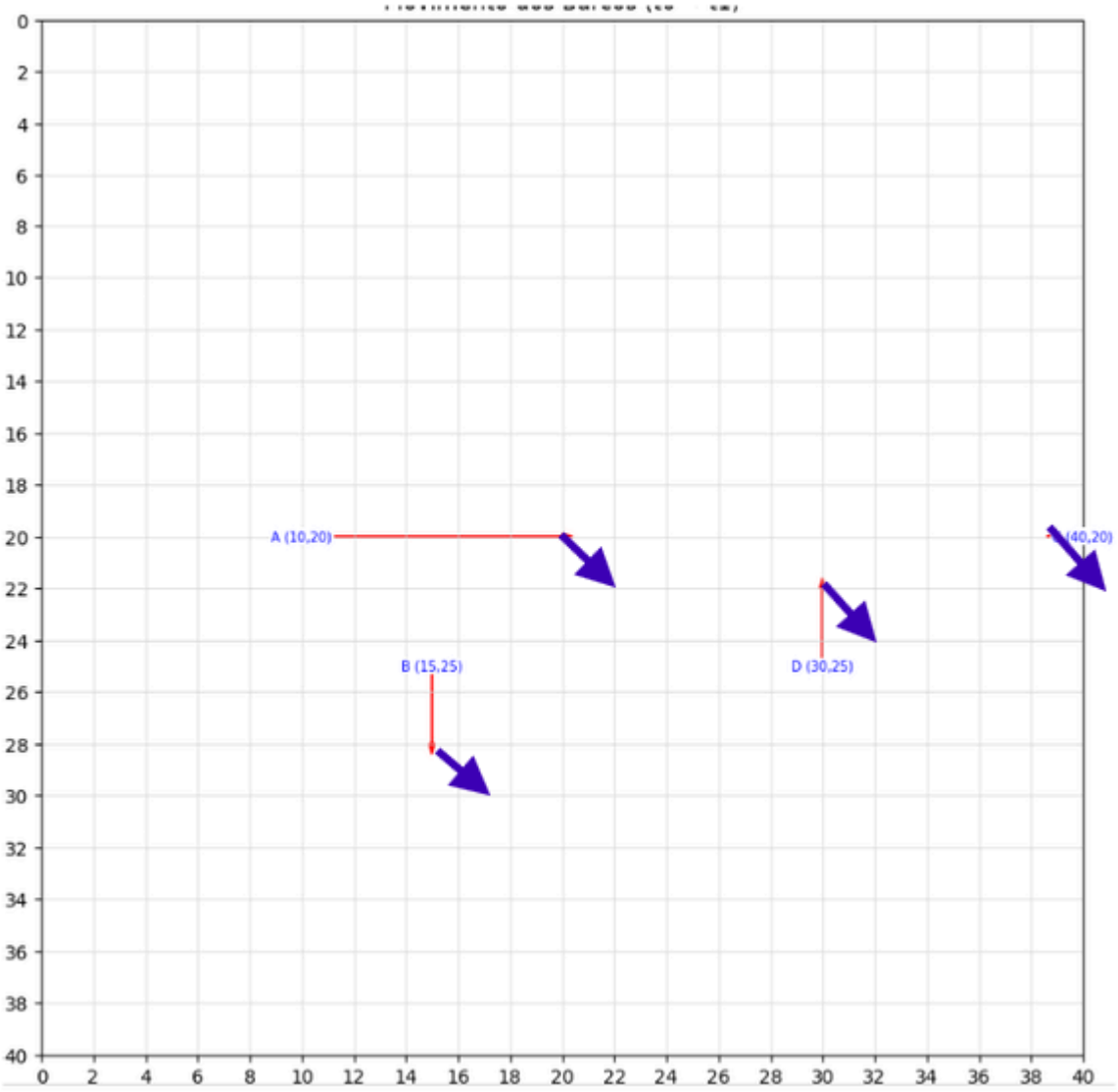
Neste caso temos uma corrente com velocidade 2 e angulo 45 graus. Reparem deslocam todos 2 casas horizontalmente e 2 casas verticalmente.

Antes:

```
A 10 20 0 10 0
B 15 25 90 3 0
C 40 20 180 1 0
D 30 25 270 3 0
```

Depois de 1 segundo:

```
A 22 22 0 10 0
B 17 30 90 3 0
C 41 22 180 1 0
D 32 24 270 3 0
```



2.9.2 Exemplo de barco normais a afundar

O barco A vai colidir com o barco B após 1 segundo, e ambos afundam.

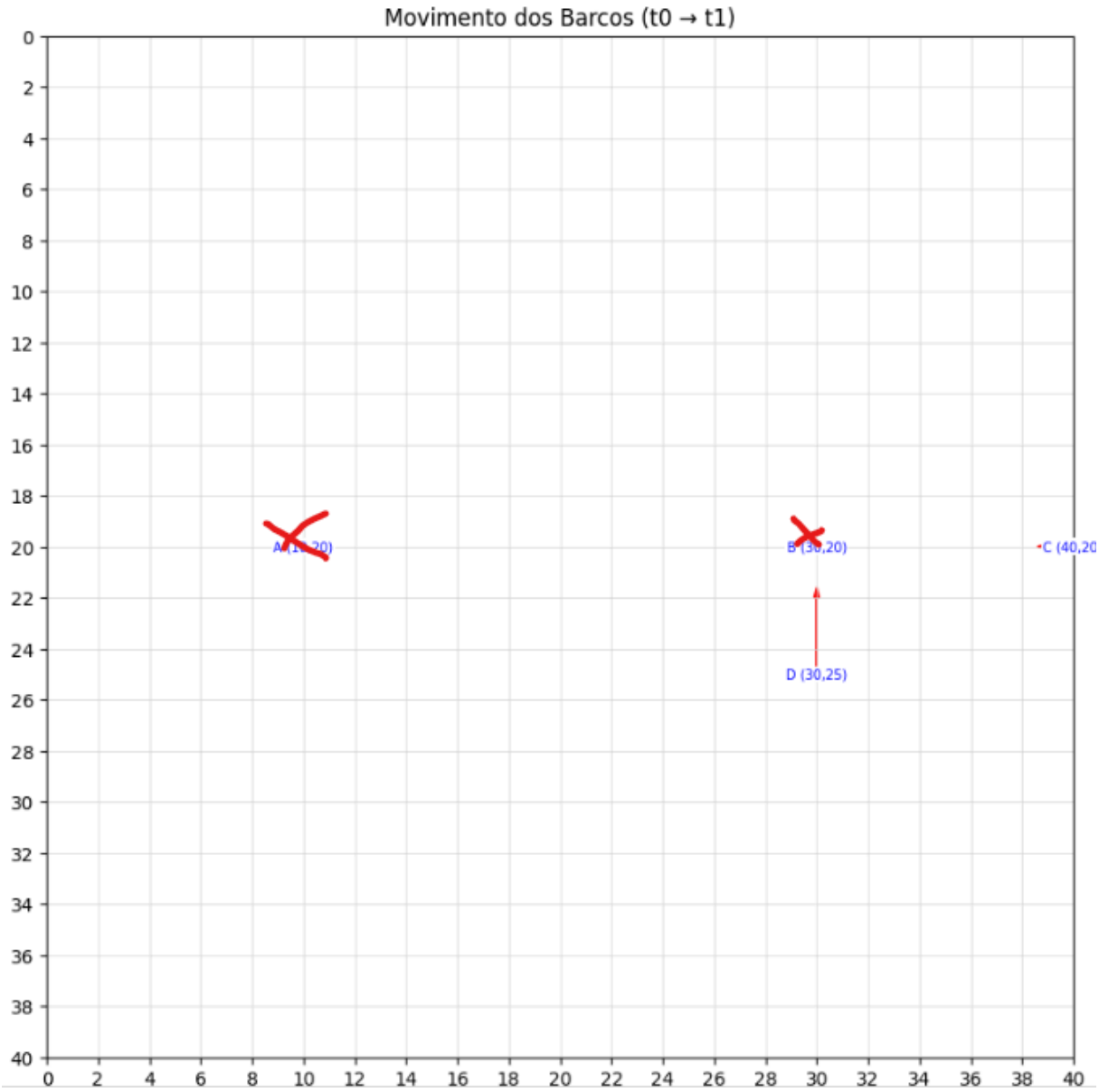
Atenção: O submarino quando não está visivel não se afunda. E afunda barco que encontra na mesma posição!

Antes:

```
A 10 20 0 10 0
B 30 20 180 10 0
C 40 20 180 1 0
D 30 25 270 3 0
```

Depois de 1 segundo:

```
C 39 20 180 1 0
D 30 22 270 3 0
```



2.9.3 Submarino

5 em 5 frames o submarino fica submerso (mas continua andar). Só pode haver colisões com um submarina quando não esta submerso.

Atenção: O Submarino não afunda quando não está visível. No entanto o outro barco na mesma posição afunda!

Antes:

```
A 10 20 0 3 2
B 20 21 180 4 0
```

Depois de 4 segundos:

```
B 46 21 180 1 2
A 28 20 0 2 0
```

Depois de 5 segundos (avancei mais 1 segundo):

```
A 30 20 0 2 0
```

Depois de 10 segundos (avancei mais 5 segundos):

```
B 40 21 180 1 2
A 40 20 0 2 0
```

2.9.4 Veleiro

Temos exemplo com 3 Veleiros.

- O Veleiro A e B vao para a direita no sentido da corrente logo ao dobro da velocidade
- O Veleiro C contra corrente logo na velocidade normal
- O Veleiro B encontra se perto do limite direito da radar, logo vai andar no sentido contrario e consequentemente anda na velocidade normal.

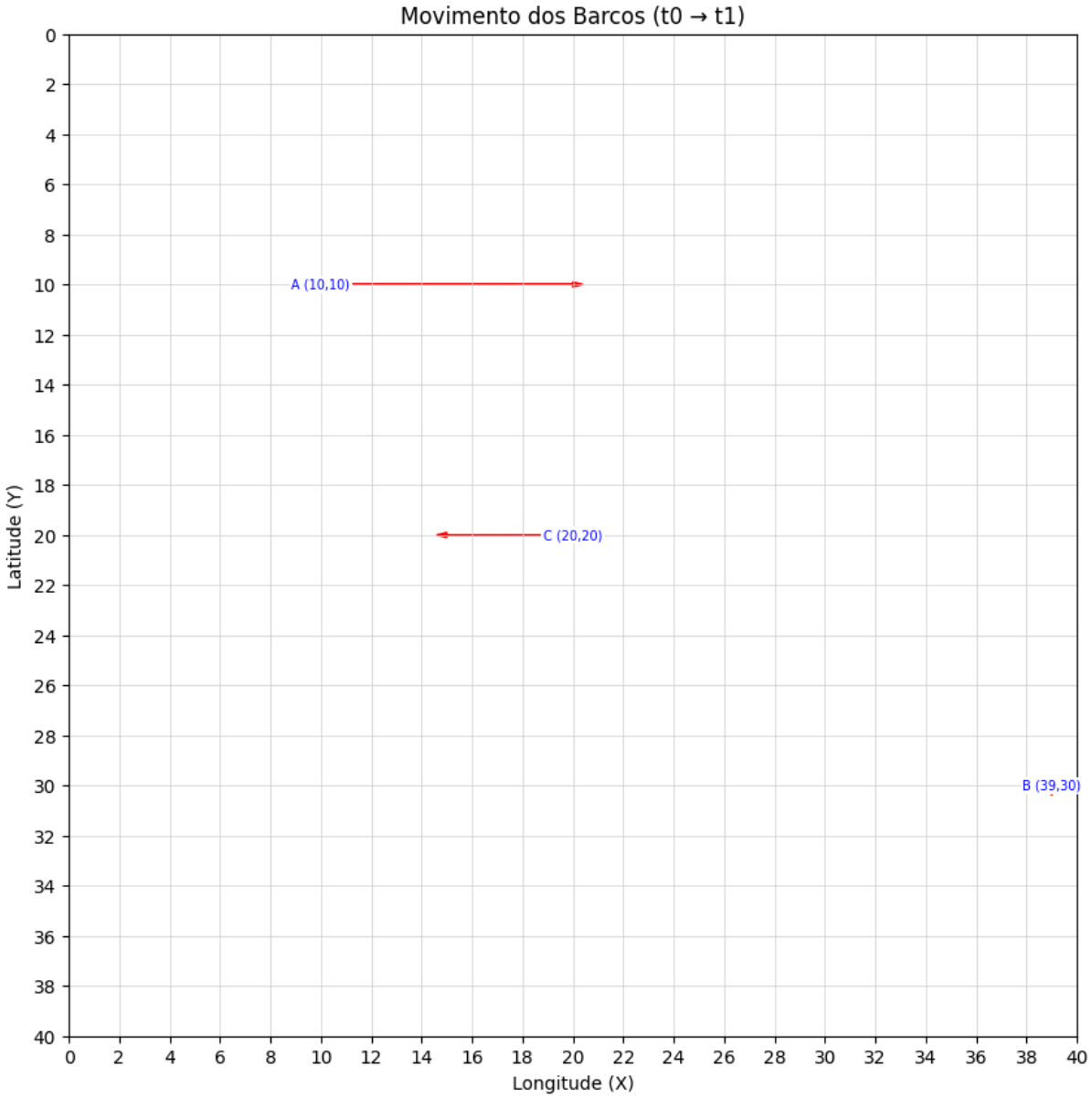
Antes:

```
A 10 10 0 5 3
B 39 30 0 5 3
C 20 20 180 5 3
```

Depois de 1 segundos:

```
A 20 10 0 5 3
B 39 30 180 5 3
C 15 20 180 5 3
```

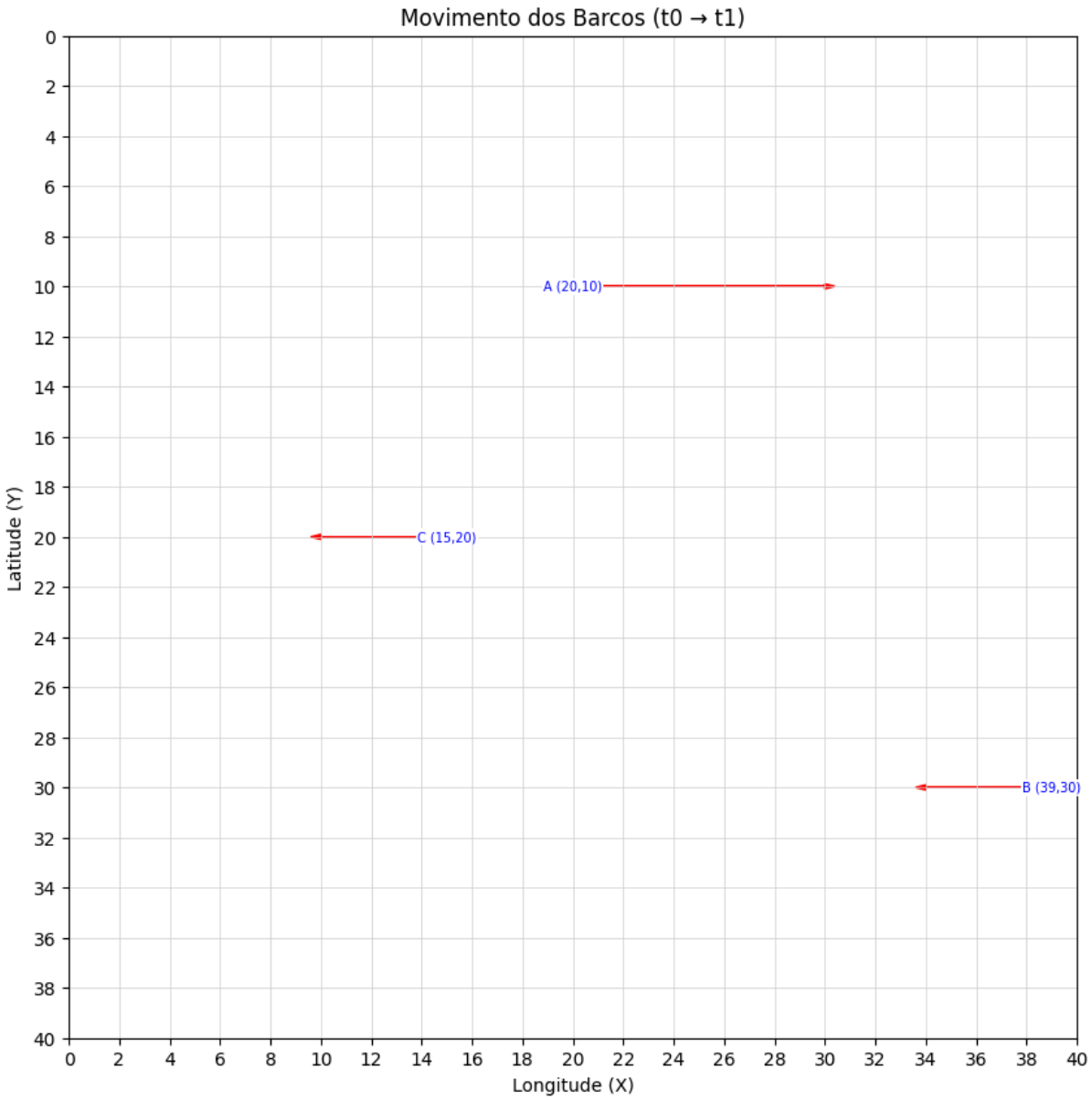
Observem que o Barco B, **manteve mesma posição** porque ultrapassou limite 40 e inverteu sentido. Atenção neste exemplo o radar é 40x40, vai de 0 a 39.



Avançando mais 1 segundo:

```
C 10 20 180 5 3
B 34 30 180 5 3
A 30 10 0 5 3
```

No segundo seguinte B já se dirige no sentido contrário.



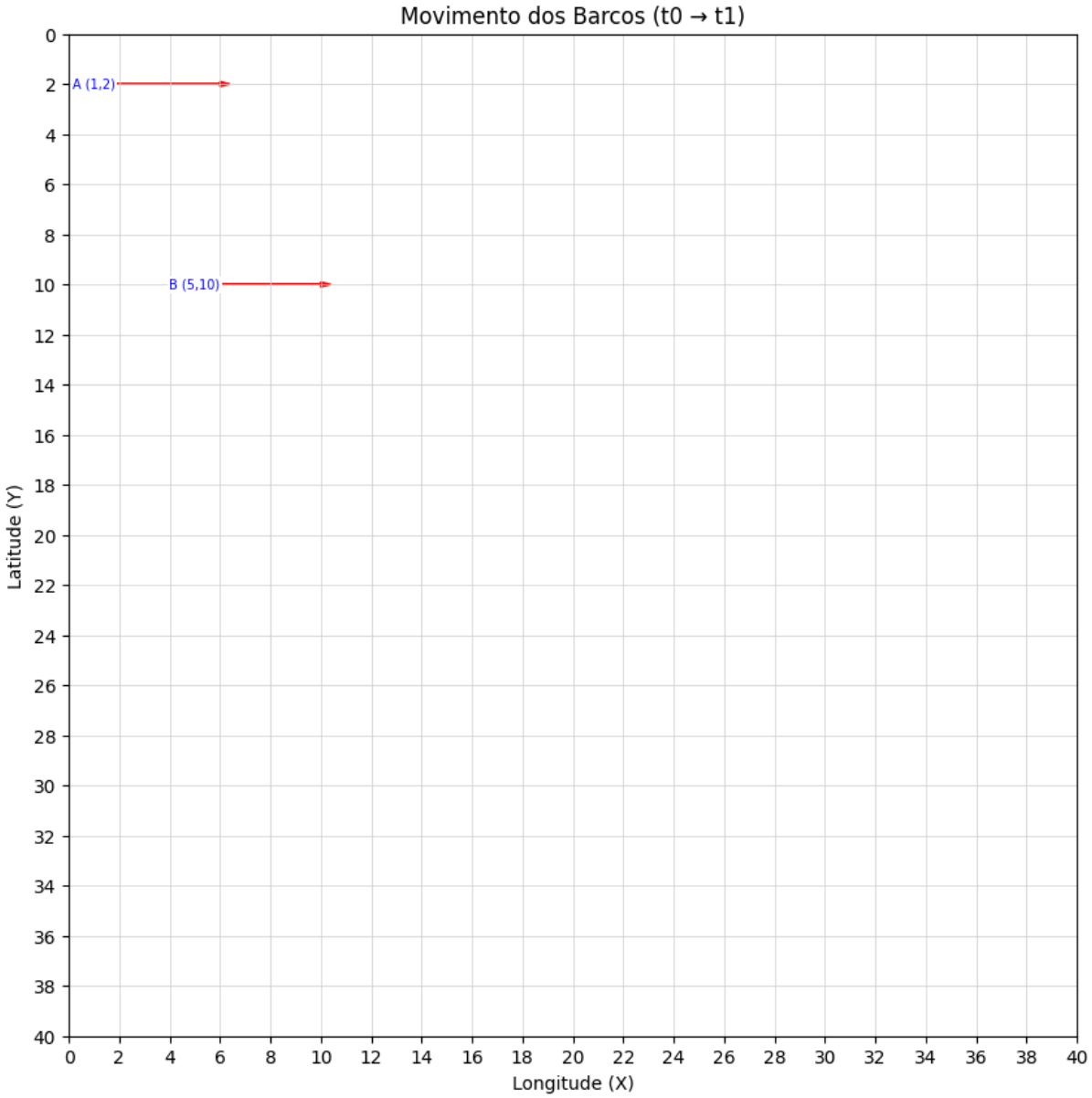
2.9.5 Drone

Antes:

```
A 20 10 0 5 1
B 5 10 45 5 1
```

Depois de 1 segundos:

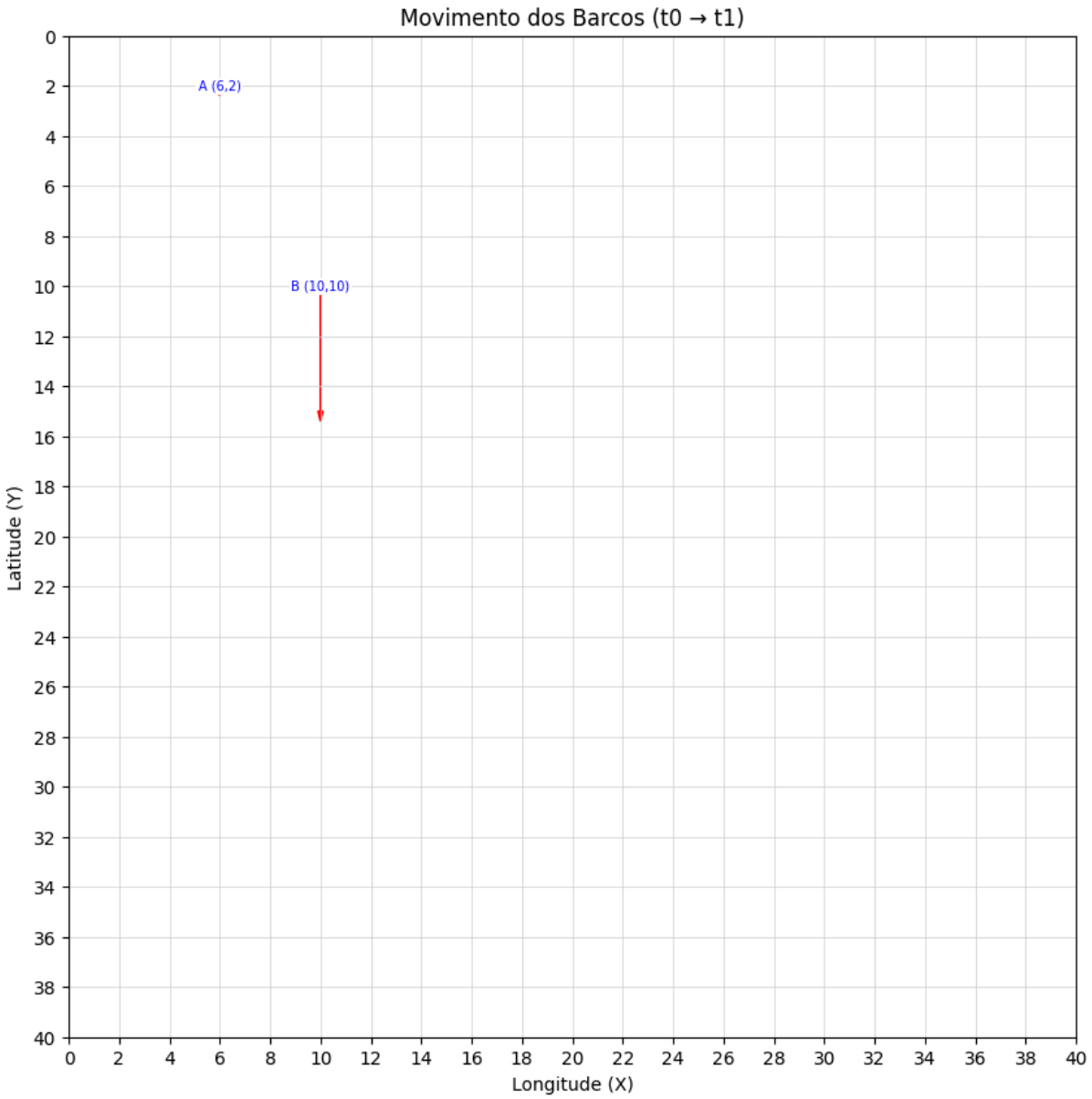
```
A 6 2 0 5 1
B 10 10 45 5 1
```



Avançando mais 1 segundo:

Observem que barco A como tem angulo 0, e nao tem movimento vertical, ele nao se move neste frame.

```
B 10 15 45 5 1
A 6 2 0 5 1
```



Apos 11 frames, o barco B está prestes a sair do radar

```
A 31 2 0 5 1
B 35 35 45 5 1
```

Barco B atinge limite horizontal, velocidade horizontal inverte, logo angulo muda de 45 para 315 graus. Reparem na posição horizontal fica no limite que é 39 (radar vai de 0 a 39).

```
B 35 39 315 5 1
A 31 2 0 5 1
```

se avancarmos mais um frame acontece o seguinte: Barco B atinge limite vertical, velocidade vertica inverte, logo angulo muda de 315 para 225 graus. Reparem na posição vertical fica no limite que é 39 (radar vai de 0 a 39).

```
A 36 2 0 5 1
B 39 39 225 5 1
```

3. Ajuda de visualização

Fica disponível um programa de visualização em Python para ajudar a visualizar o radar.

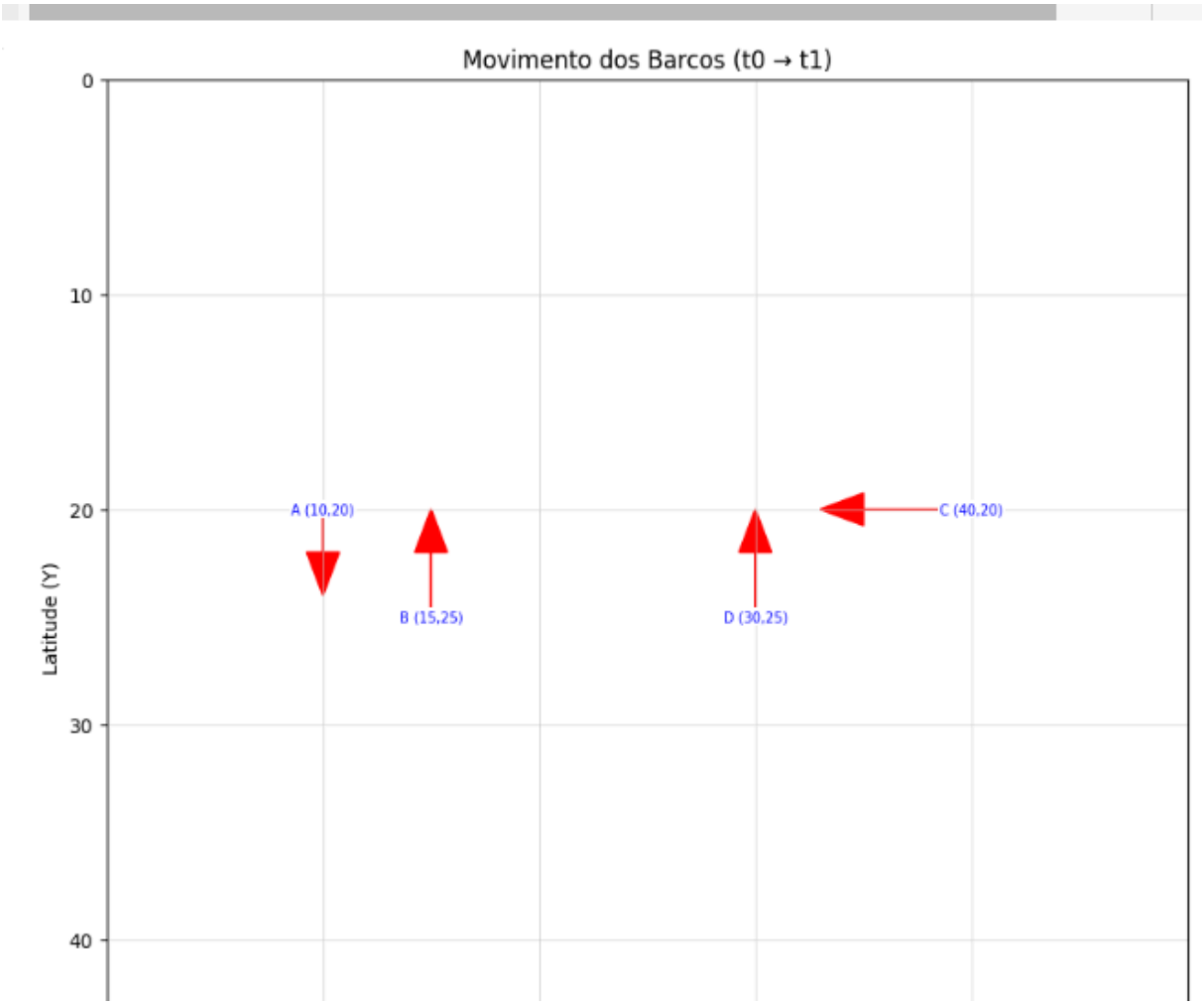
O programa lê o ficheiro de saída e gera uma imagem do radar com os barcos.

Se tiverem conta Gmail, podem usar o Google Colab para correr o programa.

O programa está em modo leitura, mas podem fazer uma cópia do programa para o seu Google Colab e executá-lo a partir daí.

https://colab.research.google.com/drive/1S7XAGY6dFyjFpCmLQlD6xxUieMN7YYsa?usp=sharing

Neste programa podem inserir o ficheiro de saída e entrada e verificar o resultado da simulação. O programa gera uma imagem do radar com os barcos. O programa lê o ficheiro de saída e gera uma imagem do radar com os barcos. Segue em baixo um exemplo do resultado, tem os mesmos barcos do exemplo de entrada, mas com a simulação a correr durante 1 segundo. As setas indicam o que os barcos percorreram durante este 1 segundo:



4. Submissão e Avaliação

A avaliação está prevista a ocorrer na primeira semana de Julho 2025. Após confirmação da reserva das salas. A data limite de entrega será informadas via email/anuncio.

Os alunos devem submeter a solução **num único ficheiro com nome `main.c`**, no moodle da DEISI no seguinte link:

https://moodle.deisi.ulusofona.pt/mod/assign/view.php?id=540

5. Critérios de Avaliação do Projeto

O projeto será avaliado em duas componentes:

1. =Nota Base=: atribuída com base na implementação entregue em C;
2. =Nota Final=: resultado da multiplicação da nota base por um fator proporcional ao desempenho nas perguntas feitas durante a defesa.

5.1 1. Nota Base (até 20 valores)

A nota base é atribuída com base na implementação das funcionalidades especificadas no enunciado. Esta parte é avaliada manualmente pelo docente.

Exemplos de funcionalidades com peso na nota:
- Opções 1, 4 e 0 corretamente implementadas, com uso de listas ligadas e ponteiros;
- Implementação das restantes opções (2, 3, 5, 6);
- Implementação dos diferentes tipos de barco.

5.2 2. Avaliação Oral com Fator de Ajuste

Durante a defesa, o aluno responderá a 3 perguntas práticas ou teóricas. Cada pergunta tem uma cotação entre 2 a 5 valores, perfazendo no total 10 valores.

A nota final será calculada como:

Nota Final = Nota Base × (Pontuação nas Perguntas ÷ 10)

5.3 3. Regras de Aprovação

Para garantir a aprovação, o aluno deve atingir uma pontuação **mínima** nas perguntas, dependendo da sua nota base:

Nota Base	Nota Mínima nas Perguntas
20	5.0
19	5.0
18	5.5
17	5.5
16	6.0
15	6.5
14	7.0
13	7.5
12	8.0
10	9.5
≤ 9.4	**Reprova**

Se o aluno não atingir o mínimo exigido para a sua nota base, será reprovado.

5.4 4. Exemplo de Cálculo

Aluno obteve nota base de **16 valores**. Na defesa, respondeu corretamente a:

- Pergunta 1 (2 valores),
- Pergunta 3 (5 valores).

Pontuação nas perguntas: **7/10** Cálculo da nota final:

Nota Final = 16 × (7 ÷ 10) = 11.2 valores

Este aluno seria aprovado, pois:

- Tem nota final superior a 10;
- Atingiu os **mínimos exigidos** (6 valores nas perguntas para uma nota base de 16).

5.5 5. Notas Finais

- Não é possível ser aprovado apenas com a entrega do projeto.
- O aluno deve demonstrar domínio sobre o que implementou.
- A defesa oral ou practica é obrigatória e decisiva para a nota final.
- O objetivo é avaliar a compreensão, não apenas a execução.

5.6 Notas Finais

- Não basta entregar o código — é obrigatório entender e saber explicar o que foi feito.
- A avaliação oral ou practica é uma oportunidade para demonstrar domínio técnico e melhorar a nota.
- O uso correto de **listas ligadas, ponteiros e memória dinâmica** é fundamental.
- A ausência de funcionalidades básicas pode limitar bastante a nota máxima possível.
- O aluno é responsavel para verificar a coreta implementação dos barcos.