Trabalho prático 3 – *Terra Incognita*

1) Informação geral

O objetivo do trabalho prático 3 é avaliar a capacidade do estudante para analisar um problema algorítmico, utilizando estruturas derivadas às apresentadas na unidade curricular, e implementar em C uma solução correta e eficiente.

Este trabalho deverá ser feito de forma autónoma por cada grupo na aula prática 12 e completado fora das aulas até à data limite estabelecida. A consulta de informação nas diversas fontes disponíveis é aceitável. No entanto, o código submetido deverá ser apenas da autoria dos elementos do grupo e quaisquer cópias detetadas serão devidamente penalizadas. A incapacidade de explicar o código submetido por parte de algum elemento do grupo implicará também numa penalização.

O prazo de submissão no Moodle de Programação 2 é 21 de maio às 21:00.

2) Descrição

Um grupo de exploradores foi enviado para um local desconhecido com o objetivo de o cartografar. Esse local é designado de *Terra Incognita*, e a missão dos estudantes de Programação 2 é desenvolver um programa que registe as informações recolhidas pelos exploradores e determine o mapa da *Terra Incognita*.

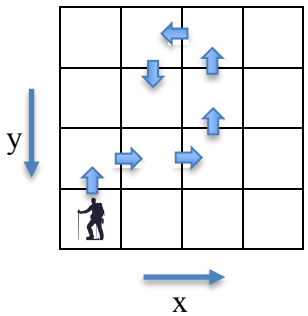
A *Terra Incognita* tem uma área retangular (ou quadrada) e é dividida em células, que são cartografadas individualmente. Cada célula é classificada nos seguintes tipos:

- Incógnita → definida por TERRA_INCOGNITA e representada por ?
-  Planície → definida por TERRA_PLANICIE e representada por _
-  Floresta → definida por TERRA_FLORESTA e representada por #
-  Montanha → definida por TERRA_MONTANHA e representada por ^
-  Água → definida por TERRA_AGUA e representada por ~

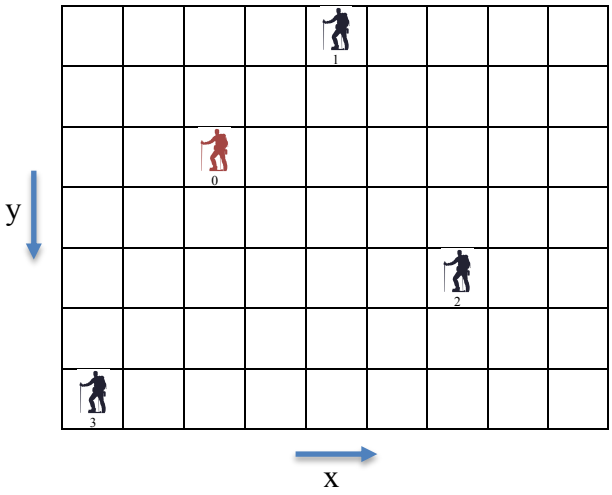
O movimento de cada *explorator* (explorador) pode ser feito em 4 direções:

- Norte ('N') → move para a célula imediatamente em cima
- Sul ('S') → move para a célula imediatamente em baixo
- Este ('E') → move para a célula imediatamente à direita
- Oeste ('O') → move para a célula imediatamente à esquerda

Por exemplo, o conjunto de movimentos N E E N N O S, resultaria no seguinte caminho:



As dimensões da *Terra Incognita* e a posição inicial dos exploradores não são conhecidas, isto é, não devem ser assumidos quaisquer valores ou restrições para estas variáveis. É, no entanto, conhecida a distância inicial de cada *explorator* em relação ao *explorator primo* (primeiro explorador, com identificador 0 no programa). Considere o exemplo ilustrado em baixo, em que o *explorator primo* está indicado a vermelho e os restantes a preto.



Neste caso, as coordenadas relativas seriam *explorator* 1 → (2, -2), *explorator* 2 → (4, 2) e *explorator* 3 → (-2, 4). É ainda importante notar que:

- Existe no mínimo 1 explorador (o *explorator primo*) e no máximo 1000.
- É possível que vários exploradores tenham a mesma posição inicial.
- Os exploradores só enviam informação sobre uma célula depois de se movimentarem.

A comunicação com os exploradores é feita através de um módulo, que já está implementado nos ficheiros `terra_incognita.h/.c`. A utilização deste módulo é obrigatória e os respetivos ficheiros não devem ser alterados. A API[1] do módulo é definida pelas seguintes funções:

- `void intro(int argc, char *argv[], int *nExplorator, int positio[][2])`

  Realiza as inicializações necessárias para a comunicação com os exploradores. Os primeiros dois parâmetros (`argc` e `argv`) são os argumentos do programa, recebidos através da função main. É devolvido por referência o número de exploradores (`nExplorator`) e as coordenadas relativas de todos os exploradores (vetor bidimensional `positio`, onde `positio[i][0]` e `positio[i][1]` indicam respetivamente as coordenadas horizontal e vertical do *explorator* com identificador `i`).

- `char explorator(int *id, int *typus)`

  Usada para efetuar comunicação com um explorador e deve ser chamada repetidamente, enquanto existirem movimentações. Retorna o movimento que foi realizado (de acordo com as 4 direções indicadas anteriormente). São ainda devolvidos por referência o identificador do explorador (`id`) que se movimentou e o tipo de terreno encontrado (`typus`). Quando não há mais movimentos a efetuar, é retornado o valor 'X'. Nota: não deve ser assumida qualquer ordem para os exploradores.

[1] https://en.wikipedia.org/wiki/Application_programming_interface

- `void tabula(cellulae_func *cell, int lat, int alt)`

  Imprime o mapa cartografado. O primeiro parâmetro é um apontador para uma função que tenha uma definição compatível com `cellulae_func` - ver explicação mais a baixo. Os outros dois parâmetros indicam a largura (`lat`) e altura (`alt`) do mapa cartografado.

- `void veritas(cellulae_func *cell, int lat, int alt)`

  Verifica a veracidade do mapa cartografado, imprimindo o número de células certas e erradas. Os parâmetros são os mesmos da função `tabula`.

- `void relinquo()`

  Finaliza as comunicações com os exploradores.

Como referido anteriormente, para imprimir e verificar o mapa cartografado, o programa deve implementar uma função com uma assinatura[2] igual a `cellulae_func`. Esta função[3] tem como objetivo indicar qual o tipo de terreno (retorno de um inteiro) que foi cartografado nas coordenadas (x, y) (dois parâmetros inteiros). É ainda importante notar que a coordenada (0,0) corresponde à célula no canto superior esquerdo do mapa da *Terra Incognita*.

Para ajudar no processo de desenvolvimento, foi criado um "esqueleto" inicial do programa no ficheiro `main.c`.

3) Avaliação

A classificação do trabalho é dada pela avaliação feita à implementação submetida pelos estudantes mas também pelo desempenho dos estudantes na aula dedicada a este trabalho. A classificação final do trabalho (T3) é dada por:

$$T3 = 0.7\ Implementação + 0.1\ Memória + 0.2\ Desempenho$$

A classificação da implementação é essencialmente determinada por testes automáticos adicionais. No caso da implementação submetida não compilar, esta componente será de 0%. Haverá uma diferenciação da classificação, de acordo com a eficiência.

A gestão de memória também será avaliada, sendo considerados 3 patamares: 100% nenhum *memory leak*, 50% alguns mas pouco significativos, 0% muitos *memory leaks*.

O desempenho será avaliado durante a aula e está dependente da entrega do formulário "Preparação do trabalho" que se encontra disponível no Moodle. A classificação de desempenho poderá ser diferente para cada elemento do grupo.

4) Submissão da resolução

A submissão é apenas possível através do Moodle e até à data indicada no início do documento. Deverá ser submetido um ficheiro *zip* contendo:

- ficheiro **main.c**
- **outros** ficheiros **.c/.h** que tenham sido utilizados como bibliotecas adicionais
- ficheiro **autores.txt**, indicando o nome e número dos elementos do grupo

Nota importante: apenas as submissões com o seguinte nome serão aceites: `T3_G<numero_do_grupo>.zip`. Por exemplo, `T3_G999.zip`

[2] Definida pelo tipo de retorno e pelo número e tipo de parâmetros.
[3] Este tipo de funções é geralmente designado de *callback*.

5) Alguns resultados esperados

Os seguintes exemplos assumem que o programa foi compilado com o nome `t3`.

```
./t3 10 150 0
=MAPA=
~^#^~^
^_~^#_
_#~_#^
#^~~~~
##_^^^
#_^_#~
======
Dimensao: 6x6
Celulas OK: 36
Celulas Mal: 0
```

```
./t3 10 50 1
=======MAPA=======
~^#???^_???????_#^
#^~?~~##???^#??_#~
_#_^#~~#??~^^??~#~
_^_##_???#~?_##_~_
~_^~^_?????__^?_^~
_##_~??????^##?^^?
_???_^??????__??__
#??_^##???????^#^~
^??^^_#???????_#^^
???????????#^?_~~_
??????????~~___^~~
??????????#_^^^##_
??????????#?~_^~_#
????????????^^~#_#
?????????????^##^#
==================
Dimensao: 18x15
Celulas OK: 135
Celulas Mal: 0
```

```
./t3 10 1000 1
=======MAPA=======
~^#^~^^_~^#__#~_#^
#^~~~~##_^^^#_^_#~
_#_^#~~###~^^_#~#~
_^_##__^_#~__##_~_
~_^~^_~#_~___^__^~
_##_~_#_#~~^##_^^_
__~#_^^#~###______
#_#_^##__##_~~^#^~
^^~^^_#^^__##^_#^^
#__^#_#_#_##^#_~~_
^~^#~~^_##~~___^~~
#^#~__~^###_^^^##_
~#_^~^#^_~##~_^~_#
_#__#^^^_~~~^^~#_#
__~~^^^_~##_~^###^
==================
Dimensao: 18x15
Celulas OK: 270
Celulas Mal: 0
```