Relatório

Até à presente data o nosso jogo encontra-se completamente funcional , isto é , executa todas as jogadas necessárias e avalia se estas são ou não válidas de acordo com as regras do jogo, realiza também o fim de jogo congratulando o jogador vencedor, e por fim, tem implementados os comandos como o gr (grava o estado de jogo num ficheiro especificado pelo jogador) , o comando Q (que termina o jogo) , o comando ler (que lê o estado do tabuleiro a partir do ficheiro), o comando Help (mostra todos os comandos disponíveis) e o comando list (que mostra os ficheiros na diretoria “saves”).

O nosso jogo possui um comando system(“clear || cls”) da biblioteca <stdlib.h> , que limpa o terminal para facilitar a leitura do jogo, porém este comando não é executado quando o jogo é corrido no CLion. Outro ponto importante é que para executar o comando gr no CLion temos de realizar uma pequena alteração que é :

- Ir a “Edit Run/debug configurations”;

- Campo “Working directory “ e alterar para a pasta principal do projeto denominada “Rastros-LI2”;

De notar que para executar este comando no terminal não é necessária nenhuma alteração. Este problema devia-se ao facto do CLion considerar a diretoria do executável a pasta “cmake-build-debug”.

Por interesse criamos compatiblidade entre sistemas operativos. Por exmplo no state/list_contents/OS_list.c usamos o #ifdef "preprocessor" para usarmos funções para ver os ficheiros na diretoria, para isso foi necessário importar diferentes módulos consoante o sistema operativo. Para tal , foi adicionado “add_definitions(-DCONFIG_CTRL_IFACE)” ao CMakeLists.txt para aceitar a syntax de #ifdef.

Adicionamos também

```
int mkdir(const char *pathname, mode_t mode);
```

no caso de ser windows para evitar warning.

Se não tiver a diretoria saves/ , no caso do comando gr , ele cria automaticamente , caso dos comandos de leitura de ficheiros (lista e ler) eles imprimem no terminal um erro de modo a alertar que tal diretoria não existe.

Em relação à IA do jogo, temos dois comandos jog, em que o primeiro jog utiliza o algoritmo flood fill para encontrar a posição mais favorável para ser jogada e o jog2 utiliza uma escolha aleatória para a jogada.O bot usa um algoritmo “Brute-Force” que testa todas as jogadas possíveis a partir de um certo ponto e escolhe, como é óbvio, a melhor.
Devido à situação presente e o facto de não termos aulas de Programação Imperativa, foi necessário o nosso esforço na pesquisa de vários conteúdos necessários à realização deste projeto.