**Relatório do Trabalho prático**

**“Ultimate Tic-Tac-Toe”**

Nome do aluno: Jorge Ricardo Marques Duarte

Número de estudante: a2021110042

Cadeira: Programação  
Curso: Licenciatura em Engenharia Informática

# Introdução

Este relatório foi realizado no âmbito da cadeira de Programação e contempla a descrição genérica da organização do programa, a identificação do ambiente de desenvolvimento utilizado durante a implementação, a apresentação de estruturas dinâmicas implementadas e da organização dos ficheiros utilizados pelo programa, as escolhas de opções tomadas em termos de implementação e ainda a especificação definida para as regras de jogo.

O programa foi implementado respeitando a norma C99 e contém uma interface simples e amigável onde é apresentado ao utilizador o contexto da jogada e os tipos de interações que pode realizar.

# Ambiente de desenvolvimento

O programa foi desenvolvido no IDE CLion, um software com suporte para as linguagens de programação C e C++. Este software tem várias funcionalidades úteis como por exemplo código *boilerplate*, assistência de código, possibilidade de *refactor* de ocorrências especificas entre outro tipo de funcionalidades.

Foi criado um repositório privado no Github que permitiu o armazenamento e progresso do desenvolvimento do jogo através de *commits* e *pushs para a branch* principal, demonstrando ser muito útil na parte de organização e de gestão de código.

# Especificação de regras adicionais do jogo

A primeira regra definida para o jogo, foi em relação à jogada inicial. A jogada inicial tem de ser definida sempre pelo jogador 1. Este jogador tem a possibilidade de escolher qual o mini tabuleiro que pretende e ainda a sua posição inicial no mesmo.

No caso de um oponente ter efetuado uma jogada numa posição (index) e o mini tabuleiro representativo do index estar finalizado, é permitido ao jogador escolher o tabuleiro e a posição que deseja colocar a sua peça.

Independentemente de ocorrer uma vitória ou um empate no final do jogo, é sempre pedido um nome para o ficheiro que será gravado e guardado com todas as jogadas realizadas durante o jogo.

# Organização do programa

## Estrutura de pasta e ficheiros

O programa foi dividido em vários ficheiros de modo a melhorar a organização do código. Dentro da pasta do programa foi criada uma subpasta game onde estão presente os vários modos de jogo, *HumanVsBot* e *HumanVsHuman*. Além disto, existem ficheiros como o player.h e player.c que contêm todas as ações que um jogador pode realizar, os ficheiros gameBoard.h e gameboard.c que contêm todas as funções associadas aos mini tabuleiros e ao jogo e por último os ficheiros utils.h e utils.c, que contêm algumas funções utilitárias utilizadas por outros ficheiros no decorrer do programa.

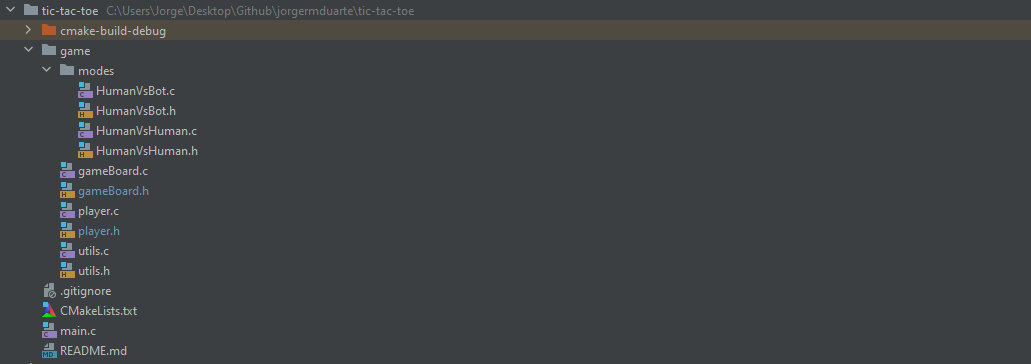


Figura - Estrutura de pastas e ficheiros da aplicação

# Estruturas da aplicação

A aplicação contém uma estrutura principal chamada *Game*. Nesta estrutura está presente o total de jogadas, um *array* contendo todos os mini tabuleiros do jogo, ponteiros para os jogadores presentes durante o jogo, um ponteiro para o utilizador que está a efetuar uma jogada no momento, o modo de jogo selecionado, uma variável booleana que representa se o jogo já acabou ou não, e uma lista ligada com todas as jogadas efetuadas durante o jogo.



Figura 2 - Estrutura principal do programa

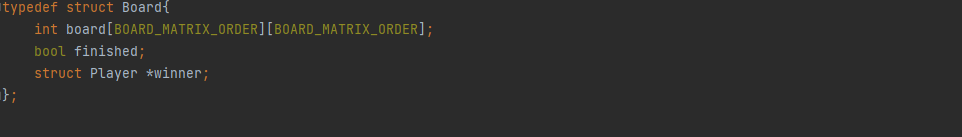
Como podemos observar na lista de mini tabuleiros (Figura 2), este tem uma estrutura específica chamada Board. Este Board contem as informações específicas do mini tabuleiro: a variável booleana que verifica se o mini tabuleiro está finalizado, o ponteiro para o vencedor, caso este exista, e uma matriz bidimensional de valores inteiros, sendo que cada valor em si representa o determinado símbolo do utilizador.

Figura 3- Estrutura de um mini tabuleiro

A estrutura Player, contem o nome do utilizador, o total de vitórias, o total de derrotas, uma variável booleana que representa se o jogador é um Bot, ou seja, um computador ou um humano e ainda o seu respetivo símbolo associado. Este símbolo é um enumerador, sendo que 0 significa NULL e é o símbolo predefinido quando é instanciado; o 1 sendo o “0” ou bola; 2 sendo o “X” e o 3 sendo “D” de empate.

Figura 4 - Estrutura de um jogador e o seu respetivo símbolo

Na estrutura Game, temos a estrutura UserPlay que é a lista ligada com todas as jogadas efetuadas durante o jogo. Esta lista ligada contém o número da jogada, o número do mini tabuleiro onde foi efetuada a jogada, o index representativo no mini tabuleiro, uma variável booleana que representa se o jogador ganhou o mini tabuleiro, outra variável booleana que representa se a jogada fez com que o tabuleiro ficasse finalizado/fechado, o ponteiro para o jogador que efetuou a jogada, o ponteiro para a próxima jogada e por último o ponteiro para a jogada anterior.



Figura 5 - Estrutura da lista ligada com todas jogadas efetuadas durante o jogo

Um dos requisitos presentes para a realização deste trabalho prático foi a necessidade de guardar o estado atual do jogo após cada uma das jogadas, permitindo aos jogadores retomarem o jogo a qualquer momento. Com esta necessidade, optei por criar estruturas para este efeito muito parecidas com as estruturas atuais, mas agora sem ponteiros. Isto permitiu guardar todas as informações necessárias para retomar o jogo.

Uma imagem com texto

Descrição gerada automaticamente

Figura 6 - Estruturas utilizadas para guardar o estado do jogo

# Especificações do programa

## Interfaces Visuais e funcionalidades

A interação visual por defeito da aplicação é a apresentação do modo de jogo desejado.

Figura 7 - Interface visual principal

Com a seleção de modo do jogo é pedido aos jogadores a introdução dos seus nomes, mas no caso do jogo ser entre um jogador e o computador, só é pedido o nome do jogador.

Uma imagem com texto

Descrição gerada automaticamenteApós a introdução dos nomes, é apresentado um estado global do jogo, os jogadores, todos os mini tabuleiros, qual o jogador que está a fazer a primeira jogada, e é questionado em qual mini tabuleiro ele quer jogar sendo a primeira jogada.

Figura 8 - Apresentação dos mini tabuleiros

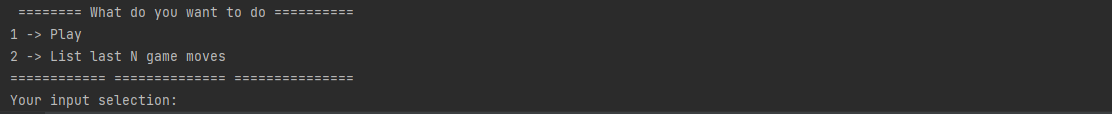
Após a seleção do primeiro mini tabuleiro o jogo entra num estado de *loop* até ser finalizado, dando a possibilidade ao jogador de jogar ou mostrar determinado número de jogadas efetuadas durante o jogo.

Figura 9 - Possibilidade jogada ou apresentação de jogadas

Uma imagem com texto

Descrição gerada automaticamenteCaso o utilizador escolha jogar é requerido o número da posição em que este quer efetuar a jogada e apresentado o estado do jogo novamente, mas agora além de todos os mini tabuleiros, é apresentado também o estado global do jogo, representativo de qual jogador ganhou ou empatou determinado mini tabuleiro. Além disto, após cada jogada, esta é guardada numa lista ligada e por sua vez utilizada para guardar o estado de jogo num ficheiro binário.

Figura 10 - Apresentação do estado global do jogo

Caso o utilizador escolha a opção para demonstrar as respetivas jogadas, é questionado quantas jogadas este quer observar, e estas são apresentadas de forma descendente, ou seja, da mais recente para a mais antiga.

Figura 11 - Lista de jogadas durante o jogo

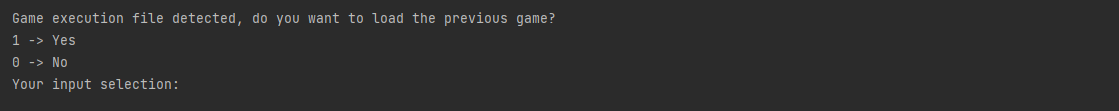
No caso de uma interrupção de jogo e se os jogadores quiserem retornar o, é possível pois após cada jogada o estado do jogo é guardado. Esta funcionalidade permite os jogadores voltarem ao estado de jogo onde tinham ficado, sempre que assim pretenderem.

Figura 12 - Carregamento do estado de jogo

Uma imagem com texto

Descrição gerada automaticamenteTendo em conta que a jogada seguinte é baseada na jogada anterior, no caso de uma jogada numa posição de um mini tabuleiro que já foi finalizado, é perguntado ao jogador em qual mini tabuleiro ele gostaria de efetuar a sua jogada além da posição da mesma.

Figura 13 - Jogada em um novo tabuleiro

Após a conclusão de vários mini tabuleiros, assim que o programa detete a vitória por parte de um jogador, é pedido na interface visual o nome do ficheiro onde desejam que todas as jogadas sejam guardadas. Estas jogadas contém as informações da lista ligada apresentando num ficheiro de texto, o número da jogada, por qual dos jogadores é que esta Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamentefoi efetuada, o mini tabuleiro a posição no mesmo.

Figura 14 - Apresentação de nome do ficheiro para conclusão final do jogo

Figura 15 - Pedido de introdução de nome do ficheiro para conclusão do jogo

Uma imagem com texto

Descrição gerada automaticamente

Figura 16 - Ficheiro de texto com todas as jogadas efetuadas

# Conclusão

Considerei este trabalho bastante interessante pois foi possível aplicar os conceitos aprendidos nas aulas teóricas, através das apresentações de exercícios práticos ao longo do semestre. Na minha opinião o mais importante foi o conceito de ponteiros e a aplicação das listas ligadas que nos levará para os próximos passos como Binary Trees, Graphs, Stacks, Queues entras outros tipos de estruturas de dados existentes e com diversos tipos de aplicações.