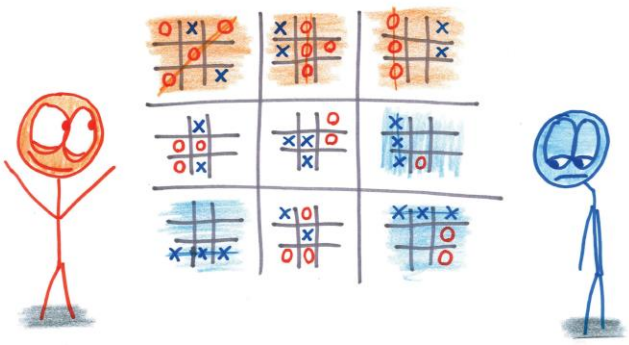
Trabalho Pratico de Programação

Docente: Francisco Pereira



Sérgio Apolinário da Costa (2020129026)

Licenciatura em Engenharia Informática – Curso Europeu

Coimbra, 2021/2022

Índice

Introdução

Este projeto surgiu no âmbito da unidade curricular de Programação durante o ano letivo de 2021/2022. Este trabalho foi feito em C standard respeitando a norma C99, e consiste na realização da versão ultimate do jogo do galo, onde existe uma área de jogo no qual estão organizados 9 mini tabuleiros do jogo do galo, dispostos numa grelha 3*3. O jogo desenrola-se entre 2 jogadores, alternado as suas jogadas, até um deles completar 3 mini tabuleiros em linha ou verificar-se o caso de empate.

Organização do projeto

Eu optei por fazer a seguinte organização do projeto criando 6 header files e 6 source files:

main.c	Contem todo o fluxo de dados, para que o programa possa funcionar como desejado.
board.h	Contem os protótipos das funções implementadas no ficheiro board.c.
board.c	Contem todas as funções necessárias para criação, manipulação e libertação de uma matriz de caracteres usando memoria dinâmica.
file.h	Contem os protótipos das funções implementadas no ficheiro file.c.
file.c	Contem todas as funções necessárias para criação e manipulação (escrita e leitura) de ficheiros binários e de texto.
interface.h	Contem os protótipos das funções implementadas no ficheiro interface.c.
interface.c	Contem todas as funções que escrevem no ecrã. Esta estruturação permite que quando for necessário alterar o texto a apresentar ao utilizador, apenas seja necessário aceder a este ficheiro, porque a informação esta toda concentrada aqui.
linkedList.h	Contem os protótipos das funções implementadas no ficheiro linkedList.c.
linkedList.c	Contem todas as funções necessárias para a manipulação de uma lista ligada simples de “*moves*”.
utils.h	Contem os protótipos das funções implementadas no ficheiro utils.c.
utils.c	Contem funções necessárias para o uso de valores gerados aleatoriamente.
struct.h	Contem todas as estruturas necessárias para a manipulação deste projeto.

Ambiente de desenvolvimento

Para o desenvolvimento deste projeto eu optei por usar o IDE CLion com a versão CLion 2021.2.3.

Estruturas e Listas Dinâmicas

Para o desenvolvimento deste projeto eu criei duas estruturas, que estão no ficheiro ***struct.h***, como referido acima.

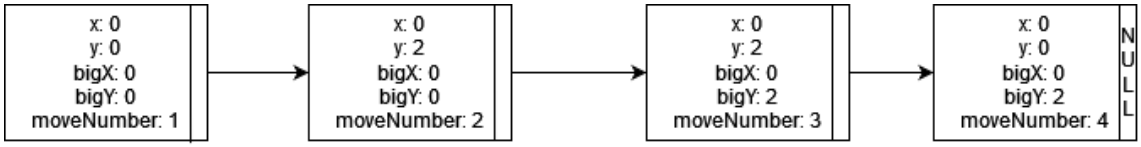
Abaixo esta representada a estrutura do jogador, que guarda como dados, o nome (***name[30]***) que pode ter até 30 caracteres, quem vai jogar primeiro (***number***), se for 1, joga em primeiro lugar, se for 2, joga em segundo lugar e como último parâmetro é guardada uma flag para identificar se o jogador é automático (***isComputer***), se tiver a 1 significa que é o jogador automático.

```
typedef struct player Players, *pPlayers;
struct player{
    char name[30];
    int number;
    int isComputer;
};
```

De seguida está representada a estrutura das jogadas, que guarda o as coordenadas introduzidas pelo utilizador (***x*** e ***y***) e as coordenadas do tabuleiro grande (***bigX*** e ***bigY***), sendo estas decrementadas um valor para facilitar a manipulação do tabuleiro. Nesta estrutura também é guardada o número da jogada realizada (***moveNumber***), e logo de seguida temos um ponteiro que vai ser usado para indicar o próximo nó da lista.

```
typedef struct allMoves moves, *pMoves;
struct allMoves {
    int x,y,bigX,bigY,moveNumber;
    pMoves next;
};
```

Na imagem abaixo esta representado um esquema da lista simples que eu criei para guardar a sequência das jogadas. Como dá para perceber nesta representação, eu optei por fazer a inserção de um novo elemento no final da lista.



Organização de ficheiros

Como foi solicitado no enunciado do trabalho pratico, quando um jogo é interrompido, guarda-se toda a informação num ficheiro binário de nome ***jogo.bin***. Quando um jogo é terminado, também se deve guardar num ficheiro de texto com o nome escolhido pelo utilizador todas as sucessões de jogadas da partida.

Para a criação do ficheiro binário eu optei por nas primeiras duas linhas guardar as informações dos dois jogadores (ordem de jogada, flag para saber se é o jogador automático e nome, respetivamente). As linhas seguintes são as jogadas normais, ordenadas, feitas pelos jogadores (número da jogada, coordenadas do tabuleiro grande (x e y) e coordenadas do tabuleiro pequeno (x e y), respetivamente). Exemplo:

```
2 0 Sergio C.
1 0 Pedro M.
1 0 0 0 0
2 0 0 0 2
3 0 2 0 2
```

Para a criação do ficheiro de texto eu optei pela mesma organização do ficheiro binário, mas desta vez com algum texto complementar, que esta a vermelho. Exemplo:

```
Play in 2 place, is computer: 0 and the name: Sergio C.
Play in 1 place, is computer: 0 and the name: Pedro M.
Move number: 1, big board row: 0, big board column: 0, small board row: 0, small board column: 0
Move number: 2, big board row: 0, big board column: 0, small board row: 0, small board column: 2
Move number: 3, big board row: 0, big board column: 2, small board row: 0, small board column: 2
```

Tomadas de decisão

Durante a criação deste projeto foi necessário tomar algumas decisões. Na lista abaixo estão descritas essas mesmas decisões:

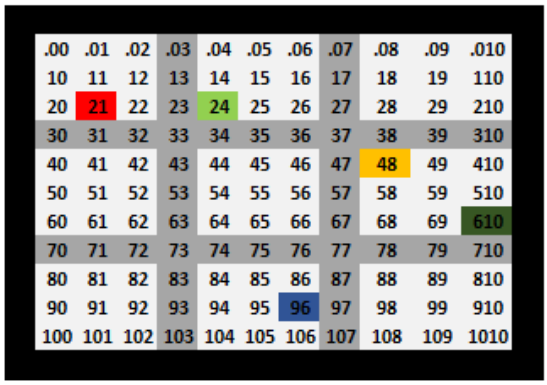
1. **Quem joga primeiro?** R: Eu optei pela escolha aleatória entre eles.
2. **Caso um mini tabuleiro escolhido para continuar o jogo já estar terminado o que acontece?** R: Eu optei por determinar aleatoriamente duas coordenadas válidas para um mini tabuleiro que esteja disponível.
3. **Onde é efetuada a primeira jogada?** R: Para não usar sempre escolhas aleatórias, eu optei por uma escolha mais básica e o jogo começa no primeiro mini tabuleiro (linha: 1, coluna: 1)
4. **Tamanho do tabuleiro e redimensionar:** Eu optei por criar uma variável global no ficheiro ***board.h*** chamada **SIZE**, em que qualquer que seja esse valor o tabuleiro vai se adaptar. Apesar de não ser pedido, eu optei por fazer cálculos para não me "prender" a tamanhos fixos.

Conclusão

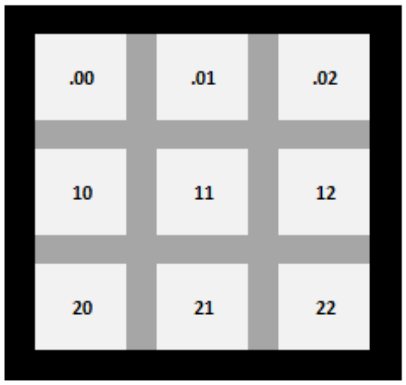
Com este trabalho pode concluir as vantagens da linguagem C, nomeadamente na manipulação de memoria dinâmica, listas ligadas, ficheiros entre outros. Além de aprender mais com estes projetos consegui um maior nível de conhecimento da linguagem.

Anexo

Em anexo, coloco uma imagem de um Excel que eu criei, para facilitar as minhas contas e conseguir chegar a uma fórmula valida e também para eu conseguir manipular o tabuleiro mais facilmente.



.00	.01	.02	.03	.04	.05	.06	.07	.08	.09	.010
10	11	12	13	14	15	16	17	18	19	110
20	21	22	23	24	25	26	27	28	29	210
30	31	32	33	34	35	36	37	38	39	310
40	41	42	43	44	45	46	47	48	49	410
50	51	52	53	54	55	56	57	58	59	510
60	61	62	63	64	65	66	67	68	69	610
70	71	72	73	74	75	76	77	78	79	710
80	81	82	83	84	85	86	87	88	89	810
90	91	92	93	94	95	96	97	98	99	910
100	101	102	103	104	105	106	107	108	109	1010



4=3*1+1	.00
8=3*2+2	.00
2=3*0+0+2	21
1=3*0+0+1	21
6=3*1+1+2	22
10=3*2+2+2	22
9=3*2+2+1	12
6=3*1+1+2	12
2=3*0+0+2	20
4=3*1+1+0	20

FormulaY=SIZE*BigY+BigY+Y
FormulaX=SIZE*BigX+BigX+X

Os BigX, BigY, X e Y devem ser tratados e começam a 0

Referências

Nome	Link	Data
stackoverflow	stackoverflow.com	Abril e maio de 2022
moodle.isec.pt	moodle.isec.pt	Abril e maio de 2022
GeeksforGeeks	geeksforgeeks.org	Abril e maio de 2022