RELATÓRIO DO TRABALHO DE PROGRAMAÇÃO

JOGO DO SEMÁFORO

CARLOS SANTOS - a2003035578@isec.pt

2020-2021

RELATÓRIO DO TRABALHO DE PROGRAMAÇÃO

JOGO DO SEMÁFORO

ALUNO N.º 2003035578

Carlos Santos

ENTIDADE

Instituto Superior de Engenharia de Coimbra

Docente responsável pela UC

Francisco Pereira

INSTITUTO SUPERIOR DE ENGENHARIA DE COIMBRA

COIMBRA – JUNHO – 2021

Notas iniciais do trabalho

O objetivo principal deste trabalho é preparar os alunos da Unidade Curricular (UC) de Programação para a implementação de soluções completas na liguagem C (norma C99). Abrindo assim caminho para o desenvolvimento de programas mais complexos.

Netes relatório irei percorrer o meu levantamento de requisitos feito a partir do enunciado, descrever brevemente o algoritmo e as principais soluções implementadas assim como a estrutura do programa.

No meu programa tentei abordar quase todo o programa desta Unidade Curricular:

- Ponteiros
- Ficheiros de cabeçalho
- Estruturas
- Ficheiros
- Estruturas Dinâmicas
- Recusividade (não implementado, matéria lecionada após finalização do programa).

ÍNDICE

1 - INTRODUÇÃO

O programa implementado é o jogo do semáforo, este é um jogo de tabuleiro entre duas pessoas que jogam alternadamente até que uma vença. O jogo do semáforo desenrola-se num tabuleiro dividido em células. No início, o tabuleiro está vazio. Alternadamente os jogadores vão colocando peças de cor Verde (G), Amarela (Y) ou Vermelha (R). Ganha o jogador que coloque uma peça que permita formar uma linha, coluna ou diagonal completa com peças da mesma cor.

O jogo realiza-se no tabuleiro, inicialmente vazio e em cada jogada, cada jogador pode realizar apenas uma das seguintes ações:

- Colocar uma peça verde apenas e só num quadrado que esteja vazio;
- Substituir uma peça verde por uma peça amarela;
- Substituir uma peça amarela por uma peça vermelha;
- Colocar uma pedra para o meio do tabuleiro (1 por jogo para cada utilizador);
- Adicionar uma coluna ou linha (2 por jogo por jogador);

As peças vermelhas e a pedra não podem ser substitudas. Se o tabuleiro não for quadrado não é posível ganhar nas diagonais e a pedra impossibilita ganhar o jogo nas linhas, colunas ou diagonais afetadas.

```
O computador introduziu as coordenadas
 L:3 C:3
    *
----------------Jogada: 8--------------

  C0    C1    C2    C3

     |     |     |
 ---   ---   ---   ---   L0
  G  |  R  |     |
 ---   ---   ---   ---   L1
     |     |     |
 ---   ---   ---   ---   L2
     |  #  |     |  Y
 ---   ---   ---   ---   L3

Jogador Um introduz as coordenadas
```

2 – Levantamento de requisitos principais

Tecnologias permitidas	O programa deve ser implementado em C standard, i.e., não deve ter instruções que o tornem específico para um determinado ambiente/plataforma de desenvolvimento. Deverá ser respeitada a norma C99.
Objectivo - Introdução	O programa a implementar deverá permitir a realização do jogo do semáforo entre 2 pessoas que efetuam jogadas altemadas, até que uma de elas vença ou que se verifique um empate.
Contexto do jogo	O jogo do semáforo desenrola-se num tabuleiro dividido em células. No início, o tabuleiro está vazio. Altenadamente os jogadores vão colocando peças de cor Verde (G), Amarela (Y) ou Vermelha (R).
Inicio do Jogo	A dimensão do tabuleiro é definida no início do jogo. O tabuleiro inicial deve ser quadrado e ter dimensão entre 3 e 5 linhas. O valor é selecionado aleatoriamente. O jogador A é sempre o primeiro a jogar.
Jogadas Válidas	As jogadas válidas relativas á colocação de peças são as seguintes: 1. Colocar uma peça Verde numa célula vazia 2. Trocar uma pela Verde por uma pela Amarela 3. Trocar uma pela Amarela por uma pela Vermelha
Jogada adicional - Pedra	Existem duas jogadas adicionais que podem ser efetuadas pelos jogadores: 4. Colocar uma pedra numa célula vazia, a pedra impossibilita ganhar o jogo nas linhas, colunas ou diagonais afetadas. Limite de uma pedra por jogo por jogador.
Jogada adicional - Linhas e colunas extra	5. Adicionar uma linha ou uma coluna ao final do tabuleiro. Esta jogada adiciona linhas ou colunas completas e vazias ao tabuleiro de jogo. Cada jogador pode efetuar esta jogada, no máximo, duas vezes por jogo.
Manutenção das Jogadas	O programa deve manter informação sobre as jogadas que forem sendo realizadas ao longo de um jogo. Esta informacão deve ser mantida numa estrutura dinâmica do tipo lista ligada.
Histórico de jogadas	Antes de efetuar o seu movimento, cada jogador pode pedir para visualizar o estado do tabuleiro nas K jogadas anteriores.
Exportação para Ficheiro	No final do jogo, a sucessão de estados do tabuleiro deve ser exportada para um ficheiro de texto, cujo nome é pedido ao utilizador. Neste ficheiro ficará a informação completa das jogadas que foram efetuadas. O ficheiro deve ser criado e escrito apenas no final do jogo, com base na informação armazenada na estrutura dinâmica que mantem as jogadas realizadas.

Interrupção do Jogo	Esta funcionalidade permite que os jogos sejam interrompidos e retomados mais tarde. O programa deve guardar num ficheiro binário, com nome "Jago.bin", toda a informação relevante que permita retomar o jogo numa altura posterior. Quando a aplicação é reiniciada deverá ser verificada a existência do ficheiro e, caso exista, o utilizador deverá ser questionado sobre se pretende continuar o jogo anterior.
Criação de um jogador Automático	Esta funcionalidade permite a um jogador humano Jogador A) realizar um jogo contra o programa. Para isso deve ser implementada uma função que simule as escolhas do jogador B. Nesta opção nao se pretende que desenvolva e implemente uma estratégia inteligente completa para o jogo. Deve simplesmente programar um jogador automático que escolha aleatóriamente uma jogada legal em cada interação.
Tipo de Interface	O programa entregue deve ter uma interface simples e amigável, indicando o que pode ser feito em cada situação. Não são valorizados programas com interfaces gráficas e/ou utilização de bibliotecas que não sejam standard, por exemplo, para usar cores ou posicionar o cursor no ecrã.
Organização dos ficheiros de código	Deve distribuir o código fonte por vários ficheiros. Para além do ficheiro com código disponibilizado com este enunciado, deverão existir, no mínimo, outros dois ficheiros com código fonte. Deverá utilizar header files para gerir as dependências entre os vários ficheiros com código fonte. Todos os ficheiros devem ter a identificação do aluno (nome completo e número), em comentário, nas linhas iniciais dos ficheiros.

3 – Algoritmo / descritivo do programa

3.1. O início – Écran de aprensentação

Explicação do jogo e as suas regras:

```
*********** O Jogo do Semaforo ***********

Inventado pelo matematico Alan Parr.

No inicio, o tabuleiro esta vazio e um tabuleiro de tamanho quadrado definido no inicio do jogo entre 3 e 5 linhas
Os jogadores colocam as pecas de cor Verde (G), Amarela (Y) ou Vermelha (R). O jogador A joga sempre em primeiro.

OBJETIVO: Ganha o jogador que coloque uma pela que permita formar uma linha, coluna ou diagonal completa com pelas da me
sma cor.

As jogadas validas sao as seguintes:
1.      Colocar uma pela Verde (G) numa celula vazia
2.      Trocar uma pela Verde (G) que esteja colocada no tabuleiro por uma pela Amarela (Y)
3.      Trocar uma pela Amarela (Y) que esteja colocada no tabuleiro por uma pela Vermelha (R)

Existem duas jogadas adicionais que podem ser efetuadas pelos jogadores:
4.      Colocar uma pedra (#) numa celula vazia. Cada jogador tem este joker que pode colocar, no maximo, uma por jogo.C
olocar uma pedra (#) inviabiliza que o jogo possa terminar por preenchimento da linha e coluna afetadas ( e, eventualmen
te tambem da diagonal ou diagonais ).
5.      Adicionar uma linha ou uma coluna ao final do tabuleiro. Esta jogada adiciona linhas ou colunas completas e vazi
as ao tabuleiro de jogo. Cada jogador pode efetuar esta jogada, no maximo, duas vezes por jogo.
```

3.2. Jogabilidade – Teclas de jogo

Mostra como jogar, as teclas possiveis de utilizar durante o jogo:

```
Como jogar:
-- Insira: digitos nas coordenadas
-- Insira:R mostra as regras            -- Insira:P Jogada da Pedra
-- Insira:M Mostra o tabuleiro          -- Insira:C Adiciona uma coluna
-- Insira:N Para novo jogo              -- Insira:L Adiciona uma linha
-- Insira:E Stats                       -- Insira:V Para verificar as ultimas jogadas
-- Insira:S Para sair do jogo
```

3.3. Estatísticas

Informação estatística do jogo:

```
Muito Bemvindo! Inicio de jogo:   09/06/2021 01:18:22

Jogos: 81       -- Movimentos: 813
Jogador um      {Ganhou 28, Empatou 6, Perdeu 28}
Jogador dois    {Ganhou 44, Empatou 6, Perdeu 28}
```

3.4. Menu principal

Menu inicial que permite escolher as opções de interação com o jogo antes do inicio do jogo:

```
Tecla 1: Jogar
Tecla 2: Regras do jogo / Como jogar
Tecla 3: Personalizar nomes dos jogadores
Tecla 4: Modo de jogo single/multi-player
Tecla 5: Apagar estatisticas
Tecla 6: Sair
```

3.4.1 Opção 0 – Jogo Anterior

Retomar o jogo anterior - Permite continuar o jogo anterior (disponível apenas se existem jogos guardados)

3.4.2 Opção 1 - Jogar

Jogar - Permite inciar o jogo num novo tabuleiro e apaga as informações do jogo anterior:

```
Tamanho do tabuleiro: 5

  C0    C1    C2    C3    C4

     |     |     |     |
 ---   ---   ---   ---   ---   L0
     |     |     |     |
 ---   ---   ---   ---   ---   L1
     |     |     |     |
 ---   ---   ---   ---   ---   L2
     |     |     |     |
 ---   ---   ---   ---   ---   L3
     |     |     |     |
 ---   ---   ---   ---   ---   L4

Jogador Um introduz as coordenadas

L:0
C:0
```

3.4.2 Opção 2 – Regras jogo

Permite visualizar as regras do jogo, é um pequeno manual de utilização do jogo.

3.4.3 Opção 3 – Nomes de jogadores

Personalizar nomes dos jogadores:

```
Indique a sua escolha:2

Nome para Jogador 1:
teste1

Nome para Jogador 2:
teste2
```

3.4.3 Opção 4 – Tipo de jogo

Escolher modo de jogo single/ multi-player – Jogador contra jogador ou jogador contra o computador:

```
Indique a sua escolha:3


Modo jogador vs computador - Alterado -> Jogador vs Jogador
```

3.4.4 Opção 6 – Apaga estatísticas

Apagar estatísticas – Apaga o ficheiro referente às estatísticas

```
Tecla 4: Apagar estatisticas

Indique a sua escolha:4

Confirme que deseja apagar as estatisticas de todos os jogos:
1 - Sim
2 - Nao
```

3.5 Decorrer do jogo

No jogo aos utilizadores vão sendo pedidos linhas “L:” e colunas “C”

```
L:0
C:0
```

A cada jogada vai aparecendo o tabuleiro no estado atual e é gravado o ficheiro “jago.bin” para recuperação dos dados caso o jogador saia a meio do programa ou ocorra algum problema, como o computador reiniciar por exemplo.

3.5.1 Modo jogador contra jogador

É pedido a cada jogador alternadamente até ao fim do jogo uma linha e uma coluna, essas entradas de dados podem também ser utilizadas para as outras interações com o jogo como descrito nas regras “Como Jogar”.

3.5.2 Modo Jogador contra computador

O computador substitui o jogador dois e vai jogando em posições aleatórias, embora algumas alterações de comportamento estejam previstas para o aumento da dificuldade. Assim sendo, foi criada a previsão de jogada antes de fim do jogo, o lançamento aleatório de uma pedra e nos tabuleiros 3x3 proteção da casa central.

3.6 Fim do jogo:

O utilizador que conseguir formar uma linha, coluna ou diagonal com as peças todas iguais ganha.

```
 Computador ganhou por linhas - Na casa: 3

----------------Jogada: 14-------------------
#1 --------> O JOGADOR 2!!! GANHOU!!! <--------
  C0    C1    C2

  Y  |  Y  |  Y
 ---   ---   ---   L0
  G  |  R  |  Y
 ---   ---   ---   L1
  Y  |     |
 ---   ---   ---   L2

Deseja gravar o jogo num ficheiro de texto?
 1 - Sim
 2 - Nao
```

De seguida, questiona-se o utilizador se pretende gravar a informação do jogo no ficheiro texto que se pretender exporta a informação do jogo.

```
Deseja gravar o jogo num ficheiro de texto?
 1 - Sim
 2 - Nao
1
Gravar ficheiro com o nome:Jogo_teste

Jogoteste.txt
```

No final, o utilizador tem três opções, começar um novo jogo, sair ou ir para o menu.

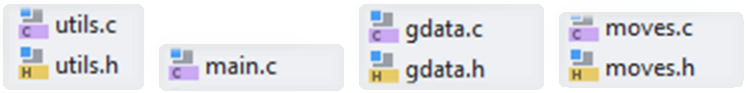
```
Deseja jogar um novo jogo?
1 - Sim
2 - Nao
0 - Parametros do jogo
```

```
0 - Parametros do jogo
0

Tecla 0: Jogar
Tecla 1: Retomar o jogo anterior
Tecla 2: Personalizar nomes dos jogadores
Tecla 3: Modo de jogo single/multi-player
Tecla 4: Apagar estatisticas
Tecla 5: Sair
```

4 – Esquema de ficheiros

O código do programa está repartido em 3 ficheiros de header e 4 ficheiros de fonte:



4.1 Ficheiros já disponibilizados “utils.c” e “utils.h”

Dois ficheiros foram disponibilizados pelos docentes para implementar no trabalho. Destes dois ficheiros utilizei a função intUniformRnd para me gerar os tabuleiros de 3 a 5 de forma aleatória na minha função no main.c e para as jogadas automáticas do computador.

4.2 O ficheiro “main.c”

Neste ficheiro acenta a estrutura de base para o jogo e inicio o jogo junto com algumas validações iniciais. É também aqui que a alocação e libertação de espaço para a lista ligada e para tabela onde guardo os movimentos de jogo são feitas. Este ficheiro utiliza algumas funções de “gdata.c” e que lança as funções de jogo em “moves.c” e onde os jogos são reiniciados.

4.3 Nos ficheiros “gdata.c” e “gdata.h”

Nestes ficheiros estão as funções mais simples, ou não relacionadas com os movimentos e que não necessitem de estruturas passadas nos argumentos das funções.

4.4 Ficheiros “moves.c” e “moves.h”

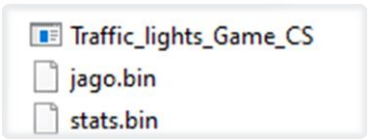
Ficheiros principais de jogabilidade e que têm algoritmo mais complexo. Aqui está a definição das estruturas que utilizei, as funções relacionadas com o “motor” do jogo e as funções que necessitem das estruturas passadas nos argumentos das funções.

4.5 Ficheiros gerados após a compilação

- “Traffic_lights_Game_CS.exe” – Programa compilado

- “Jago.bin” – Ficheiro para guardar dados de jogo

- “Stats.bin” – Ficheiro para guardar a estatísticas de jogos



5 – Estruturas

Decidi utilizar duas estruturas apenas, uma para os movimentos do jogo e informações do tabuleiro e outra para guardar os dados estatísticos.

5.1 Struct movesed – Utilização e relevância

Esta estrutura é utilizada para guardar todas as informações relevantes para jogo e é utilizada com principal relevância nas funcionalidades:

- Salvar dados de jogos para posterior recuperação em ficheiro de dados binário (função savedataend);

- Salvar dados de jogo pedido do utilizador em ficheiro de texto (função savedataend);

- Reconstrução do jogo após a entrada no jogo, se houver o ficheiro binário "Jago.bin" o utilizador derá reconstruir o jogo anterior a partir do ficheiro (função rebuild_board) ;

5.1.1 Struct movesed – Resumo da utilização no algorítmo

- A estrutura está definida no ficheiro "moves.h";

- A lista ligada é criada no ficheiro "main.c" e a cada jogada a função "insert_end" em conjunto com a função "fill_moves" inserem todas as informações relevantes à jogada na lista ligada.

- Se no início do jogo o jogador pretender recuperar o jogo anterior, a informação da lista ligada com estrutura "movesed" vai ser recuperada no ficheiro "jago.bin" e será feita a inserção de todas as jogadas salvo a última e o jogador poderá continuar o jogo a partir daí.

- Na verificação das jogadas anteriores na função "last_moves" a função "revert_piece" é utilizada para reverter as jogadas. É pedido ao utilizador o número de jogadas a visualizar e reverte-se a jogada sem introduzir na lista ligada e mostra-se o tabuleiro. Para continuar, é pedido ao jogador para carregar numa tecla e a função "last_moves_fill" preenche novamente o tabuleiro com todos os dados das jogadas até à jogada atual.

- Na função show_moves que é uma função escondida de verificação dos dados é também utilizada a lista e a estrutura, esta função pode ser chamada através da tecla "A".

- No fim de cada jogo com função "free_listed" elimina-se todos os nós da lista ligada.

```
typedef struct movesed moves, *pmoves;

struct movesed{
    int b_lines, b_cols, joker, joker_p1, joker_p2, newcolline_p1, newcolline_p2, type_player, player, game_move;
    char x, y, player1[20], player2[20], currentchar;
    pmoves ant;
    pmoves prox;
};
```

5.1.2 Struct movesed – Váriáveis e o seu propósito

b_lines	int	Guarda a dimensão da linha na jogada
b_cols	int	Guarda a dimensão da Coluna na jogada
joker	int	Pedra lançada na jogada
joker_p1	int	Número de pedras lançadas pelo jogador 1 – para limitar uso a 1
joker_p2	int	Número de pedras lançadas pelo jogador 1 – para limitar uso a 1
newcolline_p1	int	Número linhas ou colunas acrescentada pelo jogador 1 – para limitar uso a 2 por jogo
newcolline_p2	int	Número linhas ou colunas acrescentada pelo jogador 1 – para limitar uso a 2 por jogo
type_player	int	Jogador tipo (humano/manual ou computador/automático)
player	int	Guarda o jogador que fez a jogada
game_move	int	Número da jogada
x	char	Coordenada x – Linha jogada
y	char	Coordenada y – Coluna jogada
player1	char	Nome do Jogador 1
Player2	char	Nome do Jogador 2
currentchar	char	Guarda a letra que foi introduzida
ant	pmoves	Ponteiro para nó anterior da lista ligada – gestão e movimentação na lista para maior flexibilidade
prox	pmoves	Ponteiro para nó seguinte da lista ligada – gestão e movimentação na lista

5.2 Struct statisc – Utilização e relevância

Esta estrutura é utilizada para guardar informações de estatística de jogo e é utilizada com principal relevância nas funcionalidades:

- Visualizar estatísticas no écran principal e durante o jogo (função stats_read);

5.2.1 Struct statisc – Resumo da utilização no algoritmo

- A estrutura está definida no ficheiro “moves.h”;

- A estrutura é utilizada no início do para ir ler a informação ao ficheiro “stats.bin” através da função “stats_read”;

- No final de cada jogo a informação atualizada e o ficheiro “stats.bin” é criado a partir das funções “stats_create” e “update_sList”.

- Eliminação manual do ficheio de estatísticas a pedido do utilizador.

- No fim de cada jogo a função “free_slisted” elimina todos os nós da lista ligada.

```
typedef struct statisc statiscs, *stats;
struct statisc{
    char player1[20], player2[20];
    int gamesplayed,game_moves, w1, e1, d1, w2, e2, d2, type_player,winner;
    stats prox;
}
```

5.2.2 Struct statisc – Váriáveis e o seu propósito

Gamesplayed	int	Guarda o número de jogos jogados
game_moves	int	Guarda o número de movimentos totais
w1	int	Vitórias – Jogador1
e1	int	Empates – Jogador1
d1	int	Derrotas – Jogador1
w2	int	Vitórias – Jogador1
e2	int	Empates – Jogador1
d2	int	Derrotas – Jogador1
player1	char	Nome do Jogador 1
Player2	char	Nome do Jogador 2
prox	pmoves	Ponteiro para nó seguinte da lista ligada – gestão e movimentação na lista

6 – Funções

6.1 Gdata – Funções simples

No ficheiro Gdata Neste ficheiros estão as funções mais simples, ou não relacionadas com os movimentos e que não necessitem de estruturas passadas nos argumentos das funções.

void newboard	Initializa o tabuleiro com o valor: “-“
int changeboard	Para iniciar um novo tabuleiro, mas com possibilidade de escolha do tamanho
void show_board	Mostra tabuleiro - Desenha o tabuleiro de jogo
void show_board_aux	Mostra tabuleiro - Desenha o tabuleiro utilizado para fazer o revert da peça e mostrar os últimos movimentos
void show_address	Mostra o endereço de memória de cada posição de current_game no tabuleiro
int newcolline	Acrescenta linhas ou colunas a pedido dos utilizadores
int joker_play	Informações para os jogadores para o lançamento da jogada da pedra
void game_rules	Mostra as regras do jogo
void game_info	Informação inicial do jogo – Titulo do jogo e créditos
int main_menu	Menu principal do jogo
int gamemode	Alternar o tipo de jogador do jogador dois - Jogador vs jogador ou jogador vs computer
void timeprint	Mostra data e horas
void delete_stats	Elimina as informações de todos os jogos anteriores
void del_lastgame	Apaga os dados do último jogo

6.1 Moves – Funções mais complexas

Funções relativas à jogabilidade e com estruturas passadas nos argumentos.

int game_play	Função onde o jogo se passa realmente esta é a função que controla toda a parte da jogabilidade é como uma sub-main.
int valid_input	Verifica se as letras são válidas para jogar.
int valid_place	Verifica se esta dentro do tabuleiro e se não está a jogar uma peça que não pode ser substituida.
int win_count	Verifica a vitória no tabuleiro e faz uma pré verificação para o jogada automática para aumentar a dificuldade.
int place_piece	Coloca as peças no tabuleiro.
int place_piece_variant	Coloca as peças no tabuleiro para o jogador automático em caso de probabilidade de ganhar.
int revert_piece	Reverte as peças no tabuleiro, utilizada para verificar as últimas jogadas.
pmoves insert_end	Insere as jogadas no final da lista ligada.
pmoves fill_moves	Insere as jogadas no final da lista ligada em conjunto com a função insert_end.
int rebuild_board	Função para reconstruir o último jogo caso o jogador pretenda recuperar.
void show_moves	Lista de todas as jogadas na lista ligada.
void last_moves	Para verificar os últimos movimentos de jogo.
int stats_create	Adiciona as estatísticas ao ficheiro “stats.bin”.
int update_sList	Adiciona as estatísticas a partir do ficheiro “stats.bin” em conjunto com a função “stats_rebuild”.
int stats_rebuild	Adiciona as estatísticas a partir do ficheiro “stats.bin” em conjunto com a função “update_sList” e apresenta as estatísticas em conjunto com a função “show_stats”.
void show_stats	Mostra as estatísticas função chamada por “stats_rebuild”.
void free_part_listed	Liberta todos os nós da lista pmoves a partir do segundo elemento e coloca no primeiro o pmoves prox e ant a null.
void free_listed	Liberta todos os nós da lista pmoves.
void free_slisted	Liberta todos os nós da lista stats.

7 – CONCLUSÃO E NOTAS FINAIS

Foram aplicadas as técnicas apreendidas no decorrer das aulas, o programa não está completamente finalizado ou como eu pretendia mas tive que adaptar. A minha falta de experiência que originou inúmeras horas “debugging” em conjunto com a dificuldade na gestão de tempo no segundo semestre levou-me a reduzir funcionalidades, limpeza e optimização de código.

Devo ainda dizer que não conhecia o jogo e que penso que se poderá tornar bastante divertido com algumas melhorias, como por exemplo adicionar mais inteligência ao jogo, resolução de alguns bugs e acrescentar mais funcionalidades. Assim, sendo penso que o vou guardar para desenvolver mais tarde.

No entanto, foi uma fabulosa oportunidade de aprendizagem que sem dúvida alguma me permitiu consolidar e adquirir conhecimentos. Foram 255 horas de trabalho a completar este puzzle com muitas alegrias e algum despero por vezes mas que vai deixar saudades. A linguagem C ainda não é uma seguramente uma segunda língua para mim, haverei de lá chegar, vamos ver como corre a defesa do trabalho.

Muito obrigado pela atenção dispensada na correcção dos nossos trabalhos e na passagem de conhecimento durante as aulas.