U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEIC 2º Ano

Janeiro 2019

WHACK-A-TRUMP

Laboratório de Computadores

Projeto Prático

Prof. Pedro Miguel Moreira Silva

Prof. Pedro Alexandre Guimarães Lobo Ferreira Souto

Turma 5 – Grupo 7:

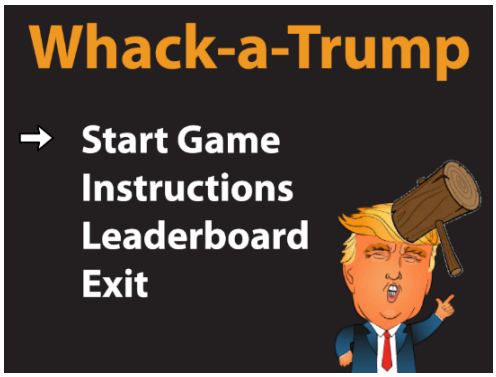
Leonor Sousa - up201705377@fe.up.pt

Sílvia Rocha - up201704684@fe.up.pt

Índice

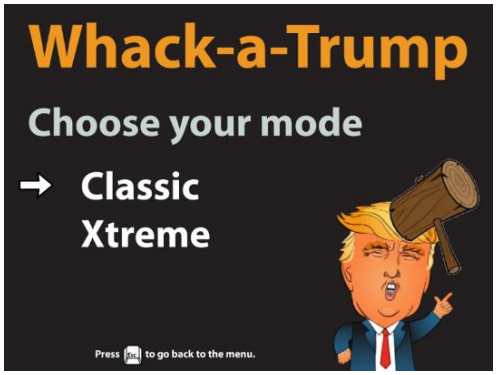
Instruções de Utilização

Menu e Submenus

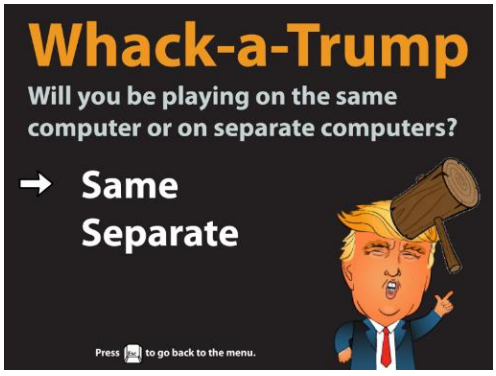


Este é o primeiro ecrã do programa. Corresponde ao menu principal a partir do qual, através do teclado, o utilizador pode escolher de entre as várias opções mostradas.

Selecionando a primeira opção (*Start Game*) surge este primeiro submenu para que o utilizador possa escolher entre o modo *singleplayer* e o *multiplayer*.



De seguida, o utilizador pode escolher entre os dois modos disponíveis: *Classic* e *Xtreme*. O submenu muito semelhante aparece quando se escolhe a opção *Leaderboard*, de maneira a ser possível escolher qual o *leaderboard* que se pretende visionar.



Finalmente, caso a opção *multiplayer* tenha sido selecionada anteriormente, surge um outro menu para escolher entre jogar no mesmo computador ou em diferentes. No primeiro caso, o jogador 1 joga no rato e o jogador 2 no teclado. No segundo caso, ambos jogam no teclado do computador respetivo.

A qualquer momento, é possível abandonar um dos submenus e retornar ao menu principal, pressionando a tecla *Esc*.

Jogo

Modo *Classic*.

Modo *Xtreme*.

Em cada um dos modos, surgem duas figuras diferentes: uma representação de uma personalidade bastante controversa na sociedade atual, Donald Trump, e uma personagem bastante conhecida e querida de toda uma geração: Granny dos Looney Tunes. A primeira figura referida, quando atingida, acresce 20 pontos à pontuação do jogador respetivo. A segunda decresce essa mesma pontuação em 15 pontos. O jogo tem a duração de um minuto sendo o objetivo atingir a maior pontuação possível nessa duração. Ao longo desse minuto o jogo intensifica-se pela redução do tempo que cada figura se mantém à superfície.

No modo *Xtreme*, como já referido anteriormente, surgem objetos que representam jornais com *fake news*. Se algum dos martelos for atingido por algum destes objetos, esse objeto é "absorvido" e a pontuação do jogador respetivo diminui em 10 pontos.

É importante referir que, no modo jogador, os dois martelos não se conseguem sobrepor. O jogador deverá ter o cuidade de não tentar efetuar um movimento que cause a sobreposição, visto que a tentativa não será bem sucedida.

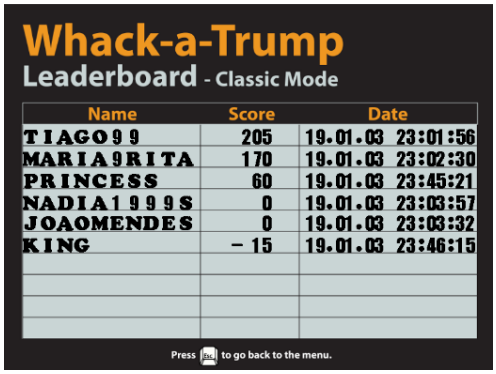
Ao longo do jogo, qualquer um dos jogadores pode colocá-lo em modo de pausa premindo a tecla 'P'. Assim que quiserem retomar o jogo basta apenas premir novamente esta tecla. Esta indicação é dada no canto inferior do ecrã de jogo.

É possível também, sair do jogo a qualquer momento, pressionando a tecla *Esc*. Contudo, se a pontuação atingida até ao momento for alta o suficiente, o ecrã seguinte surgirá de qualquer maneira.

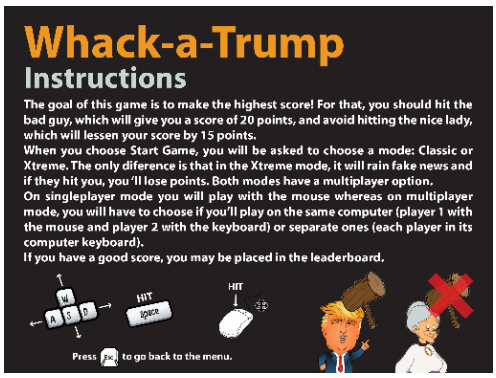
Leaderboard

Quando o jogo termina, caso algum dos jogadores tenha uma pontuação alta o suficiente, surge o ecrã aqui mostrado. Neste ecrã surge a indicação de qual/quais dos jogadores tem de escrever o seu nome para entrar no *leaderboard* do jogo. Estas informações são guardadas num ficheiro.

Eis um exemplo de *leaderboard*. Como é visível na imagem, cada jogador tem um nome, uma pontuação e uma data e hora associados. Os *leaderboards* são carregados a partir de um ficheiro no início de cada execução do programa.



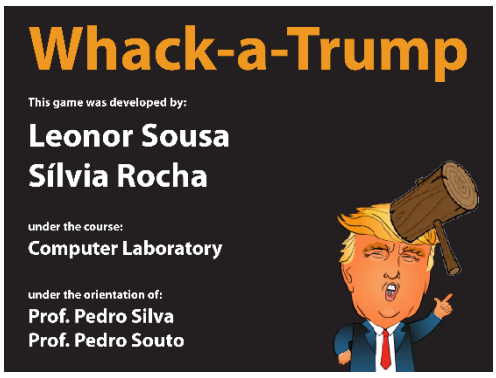
Instruções



Caso a opção *Instructions* seja selecionada no menu principal, é esta a imagem que surge no ecrã onde se explica as regras de cada modo de jogo assim como quais os dispositivos a utilizar para jogar (teclado, rato ou ambos).

Exit & Credits

Por fim, se se selecionar a opção *Exit*, surge um ecrã com os créditos do jogo. Este ecrã é fechado após 5 segundos, saindo-se do modo gráfico.



Estado do Projeto

Dispositivo	Utilização	Interrupções?
Timer	Atualização do estado do jogo.	Sim
Teclado	Interface com o jogo, navegação nos menus e inserção de nome para *leaderboard*.	Sim
Rato	Interface com o jogo.	Sim
Placa Gráfica	Desenho de imagens.	Não
RTC	Obtenção do dia e hora atual.	Não
Porta de Série	Receção e envio de dados do jogo, no modo *multiplayer* (computadores diferentes).	Sim

Timer

O timer é responsável por atualizar o estado do jogo. Tal funcionalidade inclui fazer aparecer ou desaparecer as figuras do jogo, assim como os jornais que surgem no modo *Xtreme*. Para além disso, é também usado para contabilizar o tempo de jogo. É o dispositivo base para o funcionamento do jogo. Para estas funcionalidades são usadas as funções *timer_subscribe_int()*, *timer_unsubscribe_int()* e *timer_handler().*

Teclado

O teclado é utilizado na interface do jogo, no modo *multiplayer*. Neste modo, o martelo correspondente movimenta-se consoante as teclas pressionadas pelo utilizador (teclas WASD) e é efetuada uma tentativa de pancada quando este pressiona a tecla “espaço”.

Este dispositivo é também utilizado para a navegação nos menus e submenus do programa assim como para a inserção do nome do utilizador para inclusão no *leaderboard*, quando a sua pontuação entrou no “Top 10” (input de texto).

Finalmente, é também usado na saída das várias janelas do programa (através da tecla *Esc*) e na entrada em modo pausa (através da tecla “P”).

Para que possamos fazer estas operações, usamos as funções das interrupções, *keyboard_subscribe_int()* e *keyboard_unsubscribe_int()*, e a função *kbc_ih()*. Esta última função lê o *scan code* proveniente do teclado. O deslocamento do martelo dentro do jogo é feito por uma função denominada *change_keyboard_pointer()* que muda a posição da *sprite* do cursor consoante o *scan code* recebido. Uma outra função é utilizada para testar se alguma figura

dentro do jogo foi atingida (isto é se o martelo está numa localização em que existe uma figura e tecla *space* está premida) chamada *test_for_keyboard_hit()*.

Rato

O rato é utilizado na interface do jogo. Neste dispositivo, cada movimento do rato implica um deslocamento equivalente do martelo correspondente. Para que os *packets* provenientes do rato possam ser processados são usadas as funções para ativar/desativar as interrupções e data report: *mouse_subscribe_int(), mouse_unsubscribe_int(), mouse_enable_data_report()* e *mouse_disable_data_report()*, assim como as funções *mouse_ih()* e *parse_packet()*. Para deslocar o martelo, durante o jogo, é usada a função *change_mouse_pointer()* que altera a posição da *sprite* do cursor consoante os valores de *x_delta* e *y_delta* presentes no *packet*.

Da mesma forma, um clique no botão esquerdo do rato implica uma tentativa de pancada numa figura. A função *test_for_hit()* é a responsável por determinar se existe uma pancada num local onde exista uma figura ativa e, consequentemente, o sucesso ou não dessa mesma tentativa.

Placa Gráfica

A placa gráfica é um dos dispositivos mais importantes do jogo, visto que é responsável por estabelecer a ligação com o utilizador. Assim, todo o jogo é desenhado utilizando este dispositivo. Para que isto seja possível é usada a função *vg_init()* que inicia o modo gráfico com as definições descritas abaixo.

O modo gráfico utilizado neste programa é o 0x115, com uma resolução de 800x600 píxeis, sendo que cada cor está representada no modo direto e que cada pixel possui 24 bits(RGB 8:8:8). Este modo possui 16777216 cores diferentes (2^{24} cores) disponíveis.

As animações são apenas usadas no aparecimento das várias figuras do jogo assim como dos objetos que “caem do céu”. Para o tratamento das colisões entre os martelos e entre estes e os vários jornais que surgem no modo *Xtreme*, é utilizado o método de deteção por pixel.

Neste programa implementámos também *double buffering*. Esta implementação foi feita apenas a meio do projeto pelo que conseguimos observar realmente uma melhoria na performance do nosso programa, facto que consideramos bastante positivo no processo de aprendizagem.

É também de referir a utilização, embora não da maneira mais eficiente, de *fonts* no desenho das pontuações, do tempo restante de jogo e do *leaderboard*.

Real Time Clock

O *Real Time Clock* é utilizado para ler a data e hora no momento em que um determinado jogador entra no *leaderboard* de forma a, posteriormente, guardar estes dados no *leaderboard*, junto com o nome do utilizador e a sua pontuação. Neste módulo foram, inicialmente, desenvolvidas as funções para subscrição de interrupções. No entanto, com o desenvolver do projeto, concluímos que estas funções não eram necessárias para as funcionalidades que queríamos usar. Deste modo, a função realmente utilizada deste módulo é a *read_date()* que utiliza algumas outras presentes neste módulo para que consiga ler cada um dos elementos da data e hora.

Porta de Série

Relativamente ao dispositivo porta de série, fomos capazes de desenvolver funções para *subscribe* e *unsubscribe* das interrupções correspondentes ao protocolo *Universal Asynchronous Receiver-Transmitter* (UART). Implementámos ainda algumas funções para a leitura e escrita dos vários endereços como o RBR, o THR, o IIR, o IER, entre outros. De entre estas funções destacam-se também as funções que ativam e desativam o DLAB (*Divisor Latch Acess Byte*) para que a leitura do DLL e DLM sejam possíveis. Foram, adicionalmente desenvolvidas funções para a escrita e receção de mensagens que incluem também uma função para ler o byte de confirmação de receção da mensagem. Destas destacam-se a função *write_message()*, *parse_message()* e *serialPort_ih()*.

Finalmente, foram desenvolvidas funções para configurar o UART de modo a que ambos os computadores possam estar com a mesma configuração e, desse modo, comunicarem. Nestas funções incluem-se, as funções *configure_SerialPort()* e *set_bit_rate()*.

Com recurso a estas funções, conseguimos sincronizar o arranque do jogo entre os dois computadores no modo indicado por aquele que é selecionado como jogador 1. Apesar disto, esta funcionalidade não se encontra completamente operacional uma vez que o programa termina inesperadamente sempre que tenta chamar a função *draw_sprite()*.

Organização e Estrutura do Código

files

O módulo *files* contém as funções para leitura e escrita de ficheiros. As funções para escrita em ficheiros são utilizadas para guardar e atualizar, no final de cada jogo, o *leaderboard* do modo de jogo correspondente. Já as funções para leitura de ficheiros são invocadas no arranque do programa tornando imediatamente disponíveis os *leaderboards* de ambos os modos de jogo. Deste modo, não há perda de informação entre execuções do programa.

Percentagem por membro: 10%/90% (Leonor/Sílvia)

Percentagem no trabalho: 4%

game

O módulo *game* contém todas as funções associadas ao desenvolvimento do jogo. Deste modo, possui a função game() que corresponde ao ciclo principal do jogo, as funções que atualizam os dados do jogo (pontuação, tempo restante de jogo, figuras ativas, entre outros), as funções que testam, a cada interrupção do rato ou teclado, se podem efetuar o deslocamento correspondente à informação dessa interrupção (isto é, testa se esse deslocamento origina colisões ou se colocaria o cursor fora dos limites da resolução) ou se alguma figura foi atingida.

Percentagem por membro: 50%/50% (Leonor/Sílvia)

Percentagem no trabalho: 25%

i8254

O módulo *i8254* foi-nos fornecido pelos professores aquando o desenvolvimento do lab2. Contém a definição de inúmeras constantes úteis para o desenvolvimento das funcionalidades do timer.

Percentagem no trabalho: 1%

keyboard

O módulo *keyboard* contém funções relacionadas com o manuseamento do dispositivo teclado, tais como um handler e funções responsáveis pela ativação e desativação das interrupções, entre outras.

Este módulo foi desenvolvido por nós durante a elaboração do lab3.

Percentagem por membro: 50%/50% (Leonor/Sílvia)

Percentagem no trabalho: 5%

leaderboard

O módulo *leaderboard* contém as funções relacionadas com a apresentação no ecrã das instruções e das pontuações mais altas. Possui uma função principal denominada *choose_info_menu* que, dependendo do seu primeiro argumento (menu) irá chamar a função que apresenta no ecrã as instruções ou a que apresenta o leaderboard correspondente a um outro argumento da função (mode).

Neste módulo criámos uma estrutura de dados denominada Player que contém todas as informações (nome, pontuação,data e hora) de um jogador que pertença ao leaderboard.

Percentagem por membro: 50%/50% (Leonor/Sílvia)

Percentagem no trabalho: 10%

menu

O módulo *menu* possui duas funções principais: *menu()* e *choose_option()*. A primeira corresponde ao menu principal, sendo constituída por um ciclo que permite a escolha de uma opção de entre as 4 disponíveis, devolvendo-a. Já a segunda corresponde a um ciclo para a escolha entre as duas opções disponíveis em cada um dos submenus apresentados. Uma vez mais o argumento *menu* é o que nos permite a distinção entre os vários submenus originando o carregamento da *xpm* correspondente.

Percentagem por membro: 25%/75% (Leonor/Sílvia)

Percentagem no trabalho: 5%

mouse

O módulo *mouse* contém funções relacionadas com o manuseamento do dispositivo rato, tais como um *handler*, funções responsáveis pela ativação e desativação das interrupções e de data report, assim como funções que ativam o modo de stream ou remoto e que constroem um *packet* (estrutura de dados que guarda os dados provenientes do rato), entre outras.

Este módulo foi desenvolvido por nós durante a elaboração do lab4.

Percentagem por membro: 50%/50% (Leonor/Sílvia)

Percentagem no trabalho: 6%

pixmaps

O módulo *pixmaps* inclui todos os *xpms* necessários para os gráficos do jogo, tais como, as imagens de fundo, as várias figuras do jogo (personagens, martelos e jornal) e as várias letras e números utilizados, entre outros.

Estas imagens foram, na sua maioria, desenvolvidas em ambiente Adobe Illustrator em formato *png* e, posteriormente, convertidas em formato *xpm* através do *software* GIMP.

Percentagem por membro: 50%/50% (Leonor/Sílvia)

Percentagem no trabalho: 3%

proj

O módulo *proj* é a raiz de todos os outros módulos consistindo, essencialmente, em um ciclo que mantém o programa a correr de acordo com as opções do utilizador até que este escolha sair do mesmo. Deste modo, cada iteração do ciclo chama a função menu, do módulo do mesmo nome, invocando de seguida os sucessivos submenus e funções associados às sucessivas escolhas do utilizador.

Percentagem por membro: 25%/75% (Leonor/Sílvia)

Percentagem no trabalho: 4%

rtc

O módulo *rtc* contém funções relacionadas com o manuseamento do dispositivo *RTC*, tais como funções responsáveis pela ativação e desativação das interrupções e funções que leem o dia e hora atual, entre outros.

Neste módulo foi criada uma estrutura de dados denominada *Date* onde são guardadas as informações lidas por este dispositivo.

Percentagem por membro: 35%/65% (Leonor/Sílvia)

Percentagem no trabalho: 4%

serialPort

O módulo *serialPort* contém as funções responsáveis pela subscrição e desativação das interrupções do UART assim como as funções que permitem a receção e transmissão de dados através deste canal. Entre estas funções destacam-se as funções *serialPort_ih(),* que é chamada sempre que ocorre uma interrupção desta componente e que chama a função *parse_message()* que processa os dados recebidos, e *write_message()* que é a função responsável por enviar o cabeçalho, os dados pretendidos e, por fim, o trailer byte.

Percentagem por membro: 0%/100% (Leonor/Sílvia)

Percentagem no trabalho: 8%

sprite

O módulo *sprite* contém funções relacionadas com o manuseamento de S*prites*.

Este módulo foi elaborado a partir da modificação e expansão do módulo com o mesmo nome, desenvolvido durante o lab5. Algumas das funções deste módulo foram, portanto,

adaptadas a partir das que nos foram fornecidas pelos professores da unidade curricular e pelo Prof. João Cardoso.

As funcionalidades deste módulo incluem, deste modo, a criação de *Sprites*, o carregamento de *xpms*, o desenho de *sprites* e de imagens de fundo e o teste de colisões, através da deteção por retângulo e por pixel.

Percentagem por membro: 60%/40% (Leonor/Sílvia)

Percentagem no trabalho: 5%

timer

O módulo *timer* contém funções relacionadas com o manuseamento do dispositivo timer, tais como um handler e funções responsáveis pela ativação e desativação das interrupções.

Este módulo foi desenvolvido por nós durante a elaboração do lab2.

Percentagem por membro: 50%/50% (Leonor/Sílvia)

Percentagem no trabalho: 7%

utility

No módulo *utility* constam algumas funções um pouco mais gerais como, por exemplo, funções para efetuar a correspondência entre um determinado *scan code* ou um algarismo e a *xpm* correspondente à letra/número que esse *scan code* representa.

Percentagem por membro: 50%/50% (Leonor/Sílvia)

Percentagem no trabalho: 3%

videoCard

O módulo *videoCard* contém funções relacionadas com o manuseamento do dispositivo placa gráfica, tais como funções responsáveis pela entrada e saída no/do modo gráfico e pelo mapeamento do endereço da memória de vídeo, entre outros.

É também neste módulo que são criadas as condições para que a técnica de *double buffering* tenha sido implementada.

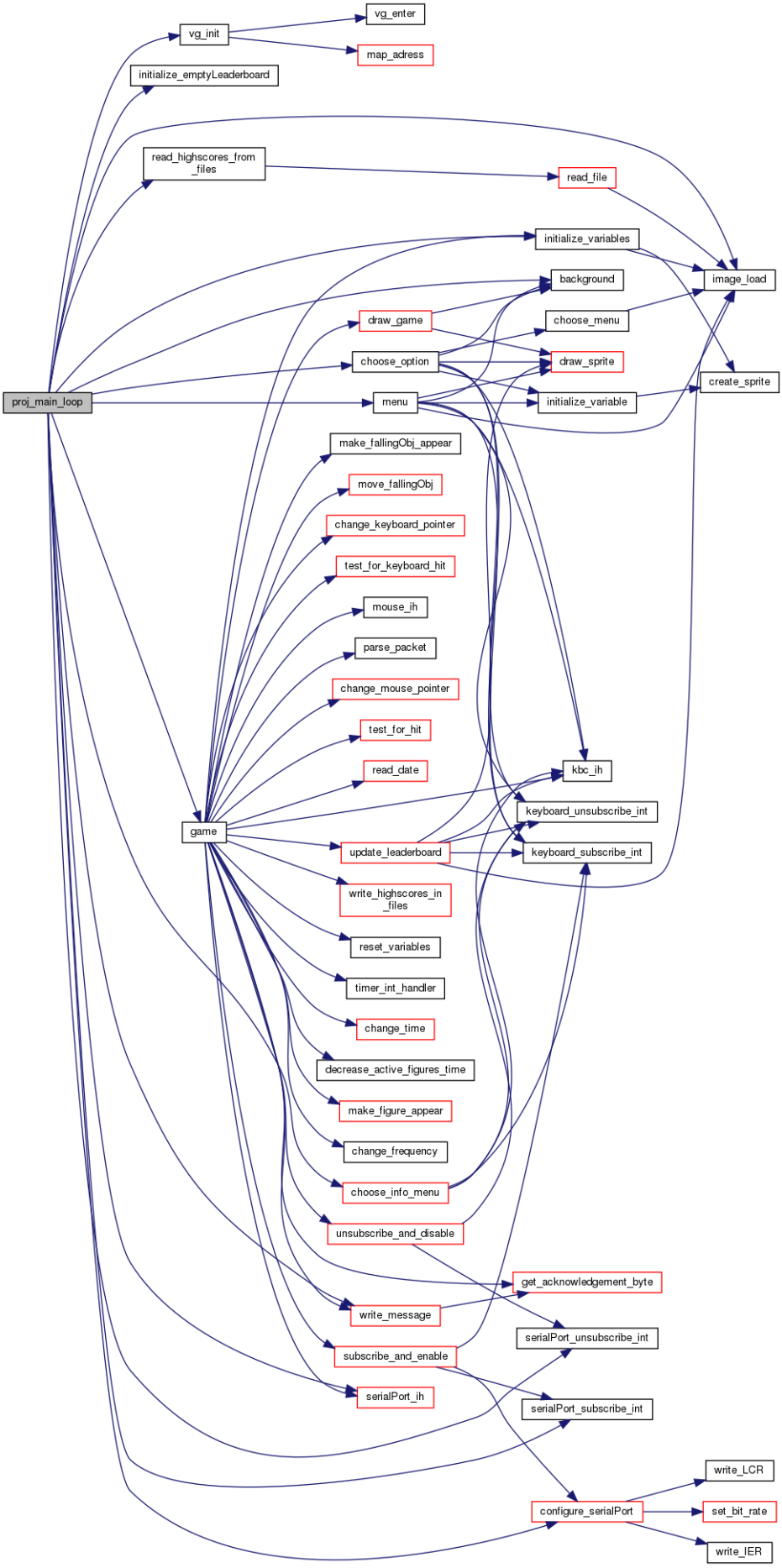
Este módulo foi desenvolvido por nós durante a elaboração do lab5, com a adição da implementação de *double buffering*.

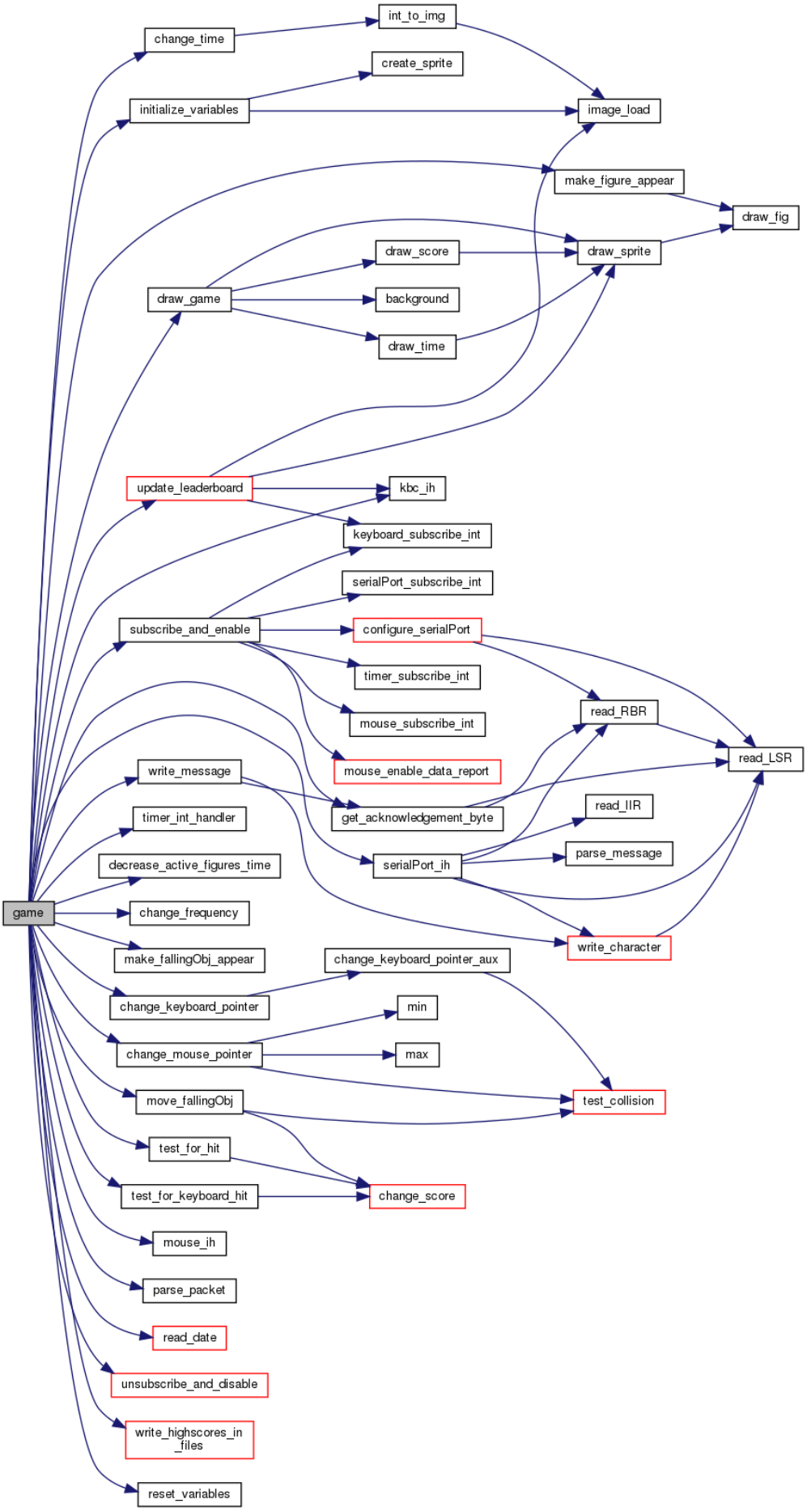
Percentagem por membro: 45%/55% (Leonor/Sílvia)

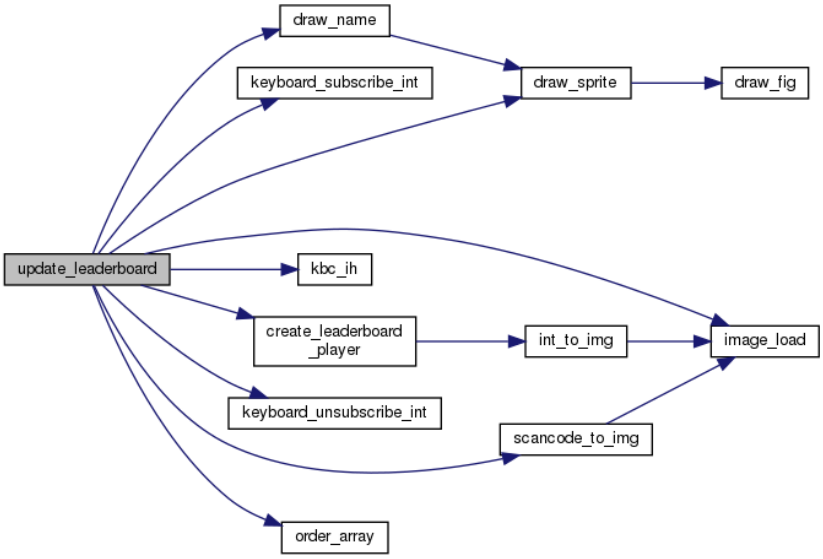
Percentagem no trabalho: 10%

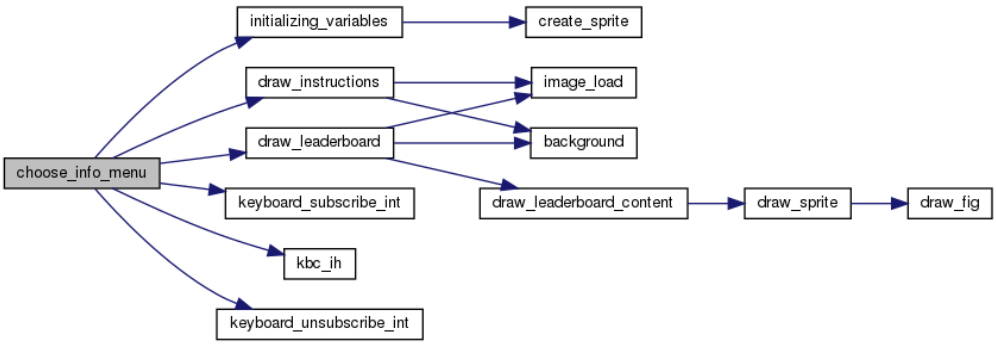
Call Graphs

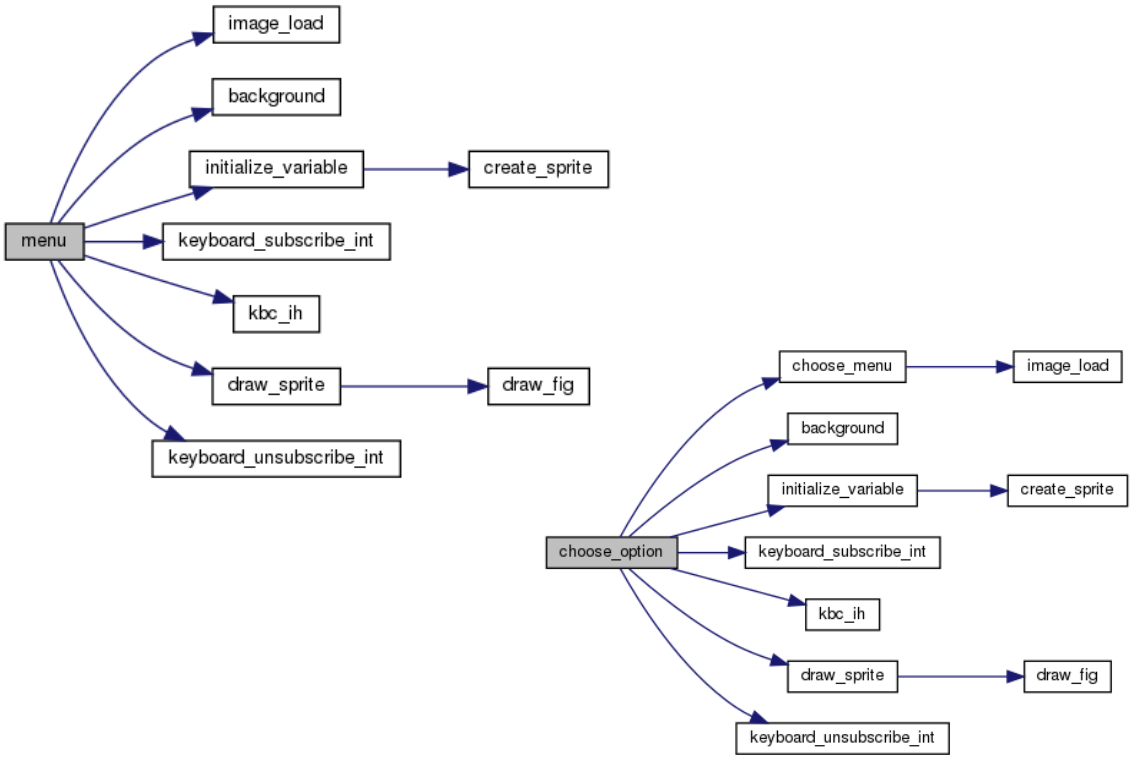
Seguem-se os *call graphs* da função principal *proj_main_loop* e de todas as funções que chamam a função *driver_receive*.


vg_enter
vg_init
map_adress
initialize_emptyLeaderboard
read_highscores_from_files
read_file
initialize_variables
image_load
background
draw_game
choose_menu
choose_option
draw_sprite
create_sprite
proj_main_loop
menu
initialize_variable
make_fallingObj_appear
move_fallingObj
change_keyboard_pointer
test_for_keyboard_hit
mouse_ih
parse_packet
change_mouse_pointer
test_for_hit
read_date
kbc_ih
keyboard_unsubscribe_int
game
update_leaderboard
keyboard_subscribe_int
write_highscores_in_files
reset_variables
timer_int_handler
change_time
decrease_active_figures_time
make_figure_appear
change_frequency
choose_info_menu
unsubscribe_and_disable
get_acknowledgement_byte
write_message
serialPort_unsubscribe_int
subscribe_and_enable
serialPort_ih
serialPort_subscribe_int
write_LCR
configure_serialPort
set_bit_rate
write_IER


int_to_img
change_time
create_sprite
initialize_variables
image_load
make_figure_appear
draw_fig
draw_score
draw_sprite
draw_game
background
draw_time
update_leaderboard
kbc_ih
keyboard_subscribe_int
serialPort_subscribe_int
subscribe_and_enable
configure_serialPort
timer_subscribe_int
read_RBR
mouse_subscribe_int
read_LSR
mouse_enable_data_report
write_message
get_acknowledgement_byte
read_IIR
timer_int_handler
parse_message
decrease_active_figures_time
serialPort_ih
game
change_frequency
write_character
make_fallingObj_appear
change_keyboard_pointer_aux
change_keyboard_pointer
min
change_mouse_pointer
max
move_fallingObj
test_collision
test_for_hit
test_for_keyboard_hit
change_score
mouse_ih
parse_packet
read_date
unsubscribe_and_disable
write_highscores_in_files
reset_variables


draw_name
keyboard_subscribe_int
draw_sprite
draw_fig
update_leaderboard
kbc_ih
create_leaderboard
_player
int_to_img
image_load
keyboard_unsubscribe_int
scancode_to_img
order_array


initializing_variables
create_sprite
draw_instructions
image_load
draw_leaderboard
background
choose_info_menu
keyboard_subscribe_int
draw_leaderboard_content
draw_sprite
draw_fig
kbc_ih
keyboard_unsubscribe_int


image_load
background
initialize_variable
create_sprite
menu
keyboard_subscribe_int
kbc_ih
draw_sprite
draw_fig
keyboard_unsubscribe_int
choose_menu
image_load
background
initialize_variable
create_sprite
choose_option
keyboard_subscribe_int
kbc_ih
draw_sprite
draw_fig
keyboard_unsubscribe_int

Detalhes de Implementação

O módulo do Real Time Clock foi implementado com sucesso. Tendo em conta o projeto que estávamos a desenvolver, não considerámos algumas das funcionalidades deste dispositivo úteis pelo que a sua não implementação não deriva de alguma dificuldade sentida mas apenas desse mesmo facto. Deste modo, este dispositivo é apenas utilizado para ler a data e hora e não na implementação de alarmes e semelhantes. Por este mesmo motivo, concluímos que não seria necessário o uso de interrupções apesar de já termos desenvolvido previamente as funções necessárias para tal (como tal estas funções encontram-se presentes no ficheiro deste módulo).

A implementação da porta de série é um caso ligeiramente diferente. Este módulo foi uma das componentes mais desafiantes deste projeto não só pela sua complexidade mas também pelo facto do seu desenvolvimento ter ocorrido já numa fase mais avançada do projeto (após o fim das aulas laboratoriais). Este facto leva a que a comunicação de ideias entre os elementos do grupo e com os docentes seja mais dificultada pelo que os resultados acabam por ser mais complicados de obter.

Apesar da maior complexidade deste dispositivo relativamente aos restantes consideramos ter compreendido o seu funcionamento. Após algumas dificuldades iniciais na organização da elevada quantidade de informação que este dispositivo implica, conseguimos perceber como configurar o UART (recorremos à primeira porta de série: COM1). Isto permitiu-nos sincronizar os dois computadores antes do arranque do jogo em si. Para que isso seja possível, existe um computador principal (correspondente ao jogador 1) e um outro (correspondente ao jogador 2) que fica a aguardar uma comunicação por parte do primeiro que lhe permita compreender quando e em que modo deverá executar o jogo. É nesta fase que surge a nossa dificuldade: o jogo inicia de forma sincronizada mas após a primeira tentativa de desenhar algo no ecrã (chamada às funções *memcpy()* ou *draw_sprite()*) o programa termina de forma inesperada em ambos os computadores. Apesar do esforço feito, a resolução deste problema não nos foi possível dentro do prazo para entrega do projeto.

Para o desenho das imagens utilizadas nos gráficos utilizamos a ferramenta Adobe Illustrator, visto ser uma ferramenta com a qual ambos os elementos do grupo se sentiam à vontade a trabalhar. Para a conversão destas mesmas imagens num formato que pudesse ser usado pelas funções desenvolvidas, utilizámos o software GIMP.

É importante referir, a implementação de funções que detetam colisões utilizando o método da deteção por píxel, assim como a utilização de double buffering e de *fonts*.

Conclusões

Laboratório de Computadores foi uma unidade curricular que nos permitiu aprender e crescer bastante pelas várias situações desafiantes em que nos colocou ao longo de todo o semestre.

Um aspeto positivo da cadeira é, claramente, o empenho e a dedicação dos docentes. Cada vez que uma dúvida surgia e contactámos algum dos docentes (quer via e-mail quer via moodle) a resposta nunca tardou. Um outro aspeto neste âmbito que gostaríamos de realçar é o método de ensino adotado. De cada vez que tivemos alguma dificuldade em compreender algum tópico ou resolver algum problema no nosso código, o docente tentou sempre explicar da melhor maneira a base teórica apontando-nos no caminho correto mas deixando algum espaço para descobrirmos as nossas respostas. Julgamos, por isso, que este método de ensino beneficia em muito relativamente a alguns outros em que ficamos a saber a resposta mas não a compreendemos totalmente.

Os atrasos na publicação dos guiões foi algo que, de facto, pensamos que terá prejudicado a nossa performance na cadeira, visto que foram algumas as vezes em que efetivamente queríamos adiantar o trabalho que tínhamos pela frente e não o podemos fazer devido à indisponibilidade dos materiais necessários.

Da mesma forma, a falta das notas ou de uma correção dos labs foi algo que pensamos que nos terá prejudicado no desenvolvimento do projeto: tivemos que confiar e reutilizar código elaborado anteriormente, sem ter a certeza de que esse código estava correto. É um aspeto que, portanto, consideramos que deve ser melhorado na próxima ocorrência da unidade curricular.

Um aspeto que também nos dificultou o percurso ao longo dos labs e do projeto, foi o pouco conhecimento que tínhamos da linguagem C. Contudo, não o vemos como, necessariamente, um aspeto negativo da unidade curricular, mas sim como um desafio que nos é proposto. Como informáticos, a capacidade de saber procurar informação é uma aptidão fundamental e esta dificuldade revelou-se como uma oportunidade para a desenvolver.

Um outro ponto que gostaríamos de destacar é a relação entre a carga horária da unidade curricular e os créditos associados à mesma. Comparativamente a outras cadeiras e àquela que é a carga horária por crédito definida (27 horas por crédito) consideramos que o tempo real dedicado acaba por ter que ser superior podendo condicionar, por vezes, o nosso desempenho nesta e noutras unidades curriculares. Apesar disto compreendemos que a redução desta carga horária implicaria o corte de alguns dos conteúdos abordados o que dificulta a tarefa de manter este equilíbrio.

Gostaríamos ainda de sugerir que em ocorrências futuras da unidade curricular exista uma aula laboratorial dedicada à porta de série. Compreendemos que a realização do lab7 completo pode não ser sustentável tendo em conta o número total de aulas laboratoriais mas consideramos que seria interessante se alguns detalhes da implementação da porta de série fossem debatidos explicados de forma um pouco mais prática do que aquilo que é possível nas aulas teóricas.

Em suma, apesar dos aspetos negativos realçados, consideramos que esta unidade curricular teve, de facto, impacto na nossa formação e desenvolvimento não só como informáticas mas também como engenheiras pela forma de pensar que nos foi incutida. Com bastante esforço e trabalho, bastantes dificuldades também ocasionalmente, consideramos ter conseguido ultrapassar este desafio de forma bastante positiva.

Apêndice – Instruções de Instalação

Chamada do Programa

Existem duas maneiras de chamar o programa: lcom_run proj e lcom_run proj <número do jogador>, em que <número do jogador> deverá ser 1 ou 2.

Utilizando a primeira forma, obter-se-á um programa totalmente funcional, mas que não permite o modo multijogador em computadores diferentes.

A segunda forma, permite este modo, apesar de não estar finalizada e, portanto, não funcionar corretamente. O argumento <número do jogador> deverá ser 2 caso seja o jogador 2 do modo que usa a porta de série e 1 caso contrário (isto é, caso seja o jogador 1 do modo que usa a porta de série ou queira correr o programa sem recurso a este dispositivo).

Utilização de Ficheiros

Como referido anteriormente, o nosso programa guarda os dados do *leaderboard* em dois ficheiros distintos: *highscores_classic.txt* e *highscores_extreme.txt*.

Na elaboração desta funcionalidade, dado que não conhecíamos nenhuma outra forma de o fazer nesta linguagem, tivemos de atribuir, na abertura dos ficheiros, um *path* específico. Deste modo, estes ficheiros são automaticamente criados na seguinte localização: */home/lcom/labs/proj/highscores_<mode>.txt*.

Caso seja necessário alterar este *path*, deverá aceder-se ao ficheiro *files.c* nas linhas 20, 21, 103 e 104.