# LCOM Project Report

PACNIX – The arcade Pac-man game in the Minix system


Universidade do Porto

Faculdade de Engenharia

FEUP


Mestrado Integrado em Engenharia Informática e Computação – 2º ano – 2014/2015

Turma 3, Grupo 5


André Sousa Lago (up201303313@fe.up.pt)

Leonardo da Silva Ferreira (up201305980@fe.up.pt )

# Índice

# Introdução

Este trabalho foi feito no âmbito da disciplina Laboratório de Computadores, do 2º ano do Mestrado Integrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto.

O objetivo do trabalho era construir um programa com base em C e Assembly que utilizasse o maior número de periféricos possíveis (timer, teclado, rato, …), numa versão do sistema operativo Minix, adaptada à unidade curricular LCOM.

Como tal, propusemo-nos a realizar uma versão alterado do jogo clássico Pacman chamada PACNIX, que trouxesse as funcionalidades típicas do jogo original e que implementasse também outras novas, como a possibilidade de um jogo em multiplayer.

Neste relatório iremos explorar as principais funcionalidades do PACNIX, bem como demonstrar o seu método de funcionamento. Para além disso, vamos explicar quais os objetivos do trabalho cumpridos e descrever as ferramentas que utilizamos no desenvolvimento do programa.

# Instruções para o utilizador

Em primeiro lugar, operações com o jogo devem ser feitas com acesso root ao sistema. Para isso, antes de avançar para a instalação ou execução do programa, deve ser utilizado o comando "su" para garantir as permissões necessárias ao programa.

Para instalar o jogo, deve-se navegar para a pasta "proj" do projeto, e executar os scripts de instalação e compilação (por esta ordem), usando os comandos "sh install.sh" e "sh compile.sh".

Para iniciar o jogo, basta, na mesma pasta, executar o script de execução, com o comando "sh run.sh".

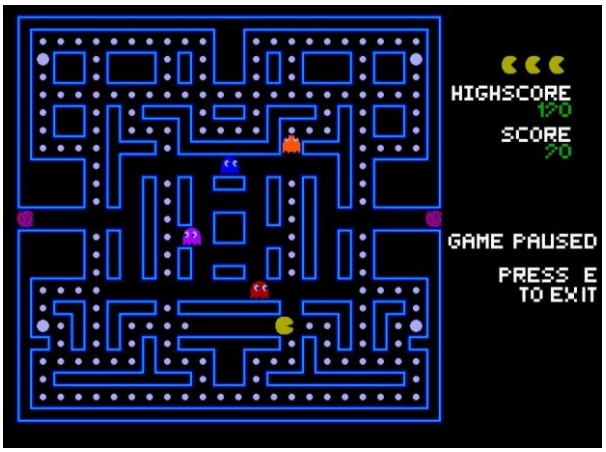
## Menu principal

Fazendo isto, surge na consola um ecrã com o aspeto em baixo representado. Este é o menu principal do jogo. Este pode ser navegado com o rato ou com as setas para cima e para baixo do teclado, podendo selecionar a opção pretendida tanto com um clique do rato como premindo a tecla "ENTER".


Segue a descrição das ações das opções do menu:

- **Singleplayer** – inicia o modo do jogo para apenas um utilizador. Quando o jogo é terminado, passa a aparecer no ecrã a pontuação obtida no último jogo realizado, se este tiver sido em modo singleplayer.
- **Multiplayer** – inicia o modo do jogo para dois jogadores
- **Instructions** – mostra um ecrã com as instruções/controlos do jogo, para os modos singleplayer e multiplayer
- **About** – mostra um ecrã com os créditos do jogo, incluindo o contexto da realização do jogo, os seus autores e alguns agradecimentos.
- **Exit** – termina a execução do programa

## Modo Singleplayer




Neste modo de jogo apenas é possível um utilizador, que deve controlar o Pacman com as setas do teclado de forma a "comer" todos os pontos e Energizers, evitando ser alcançado pelos fantasmas. O jogador possui três vidas, que vai perdendo sempre que os fantasmas o alcançarem. O jogo termina quando não existirem mais pontos ou energizers ou quando o Pacman fica sem vidas.

É também possível pausar o jogo premindo a tecla "Esc". Uma vez pausado, é possível retomar o jogo com a mesma tecla ou sair deste usando a tecla "E".

Quando o Pac-man consome um energizer, os fantasmas tornam-se todos azuis e passam a fugir dele. Enquanto os fantasmas estiverem azuis (ou brancos, como é explicado mais à frente), o Pac-man pode "matá-los", ganhando pontos por isso. Quando um fantasma morre, ele volta para a sua posição original, deixando de fugir do Pac-man. O Pac-man apenas pode comer os fantasmas durante um curto período de tempo. Quando este tempo se aproxima do fim, os fantasmas piscam entre as cores azul e branco, regressando depois ao seu estado original.

Para além de tentar "comer" todos os pontos e energizers, o utilizador pode capturar bónus de pontuação. Estes são frutas que aparecem no mapa de jogo, e que uma vez capturados dão mais pontos ao utilizador.

No final do jogo, se o Pac-man ganhar, ele recebe um bónus de pontuação por cada vida que não tiver perdido.

Nota:

O pretendido em relação ao movimento do Pac-man é que, uma vez movendo-se numa direção, ele segue nessa direção enquanto não tiver obstáculos pela frente, mesmo que o utilizador não esteja a premir nenhuma tecla. Se o utilizador premir uma tecla que implique uma mudança de direção, o Pac-man irá mudar de direção apenas quando/se puder, isto é, tiver um caminho livre, facilitando a viragem nas esquinas do mapa e evitando que o Pac-man fique virado parado contra uma parede sempre que o utilizador indicar uma direção "bloqueada".

## Modo multiplayer

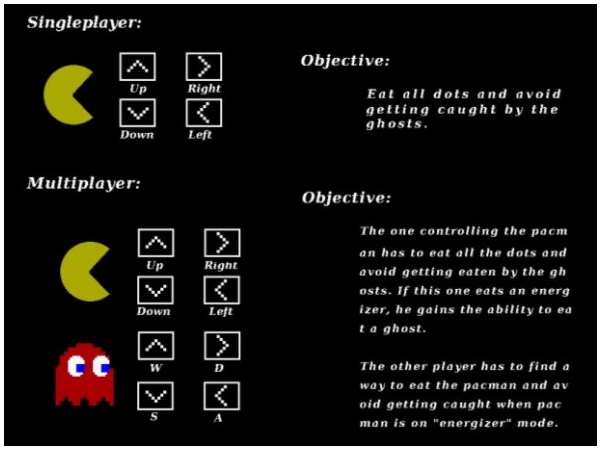
Este modo permite que dois utilizadores joguem simultaneamente e de forma competitiva.

Um dos utilizadores controla o Pacman com as setas do teclado, tendo o mesmo objetivo do modo singleplayer.

O outro utilizador pode utilizar o rato para selecionar um dos fantasmas, passando a controlá-lo (é possível alternar entre fantasmas clicando no pretendido, e desseleccionar o fantasma atual clicando numa zona vazia, isto é, sem fantasmas, do ecrã). Quando seleciona um fantasma, este utilizador passa a controlá-lo com as teclas "W", "A", "S", e "D" do teclado, tendo como objetivo alcançar o Pacman antes de este terminar o jogo. Cada vez que ele (ou um fantasma selecionado) capturar o Pacman, este perde uma vida.

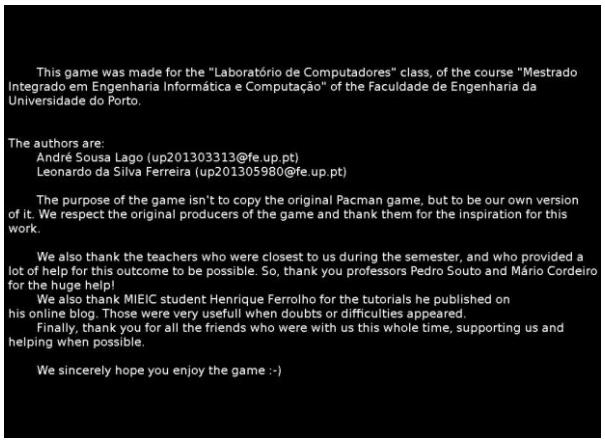
Este modo de jogo não considera a pontuação do Pacman nem dos fantasmas.

## Instructions



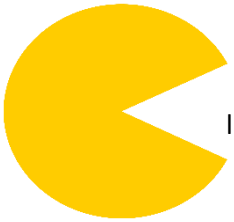

Este ecrã mostra ao utilizador os controlos do jogo para os modos singleplayer e multiplayer, como é visível na imagem. Para além disso, é também explicado o objetivo de cada um dos modos. É possível sair deste ecrã premindo a tecla "Esc" do teclado.

## About




Aqui estão presentes os "créditos" do jogo. Para além do contexto da realização deste, estão referenciados os autores do jogo bem como outras pessoas que contribuíram de alguma forma para a realização do projeto. É possível sair deste ecrã premindo a tecla "Esc" do teclado.

## Sobre o jogo:



### Mapa

O mapa do jogo é constituído por paredes azuis, pontos (pac-dots), energizers, pontos de teletransporte e bónus. As paredes constituem os limites do campo, sendo que nem o Pac-man nem os fantasmas as conseguem atravessar. Os pac-dots são um dos objetivos do Pac-man, uma

vez que capturá-los permite vencer o jogo e ganhar pontuação. Por defeito, a pontuação dos pac-dots é de 10 pontos.

O outro objetivo do Pac-man são os energizers. Quando consome um, o Pac-man ganha 50 pontos e passa a poder matar os fantasmas, durante um curto período de tempo sendo que, comendo um fantasma ganha 300 pontos. Quando este período chega ao fim, os fantasmas regressam ao comportamento que tinham antes do energizer ser consumido.

Os pontos de teletransporte, situados na lateral do campo, permitem que tanto Pac-man como fantasmas vão de um lado para o outro do campo. Este são, por vezes, úteis para o Pac-man quando tenta fugir dos fantasmas.

Os bónus aparecem por vezes no mapa sob a forma de frutas. Se o Pac-man os capturar durante o pequeno período de tempo em que estão disponíveis, ele tem um aumento de 200 pontos na sua pontuação. Para além disso, se o Pac-man ganhar o jogo, ele recebe um bónus de 200 pontos por cada vida que não tiver perdido.

## Fantasmas

Os fantasmas são criaturas que vagueiam o campo em busca do Pac-man. Quando não estão a ser controlados por nenhum utilizador, os fantasmas possuem 3 modos básicos de ação: aleatório, busca e fuga. Segue a descrição destes modos:

- **Aleatório** – neste modo, os fantasmas seguem uma direção até que surja um obstáculo ou haja uma nova direção possível de seguir (exceto a direção oposta àquela em que se dirige, isto é, numa "corredor" em linha reta, os fantasmas nunca vão para a direção de onde vieram). Quando ocorre uma destas situações, o fantasma analisa todas as direções em que se pode deslocar escolhendo aleatoriamente uma delas.
- **Busca** – neste modo, os fantasmas seguem a direção mais direta para o local onde se encontra o Pac-man, desde que esta não se encontre obstruída. Para fazer isto, o fantasma utiliza as coordenadas do Pac-man para saber em que direção ele se encontra, seguindo uma rosa-dos-ventos com 16 eixos (N, NNE, NE, ENE, E, …). O comportamento deles segue o seguinte padrão: se o Pac-man está na direção NNE, vou tentar ir para norte (cima). Se esta direção estiver obstruída, vou para E (direita) que é a direção mais próxima do Pac-man, sem ser Norte. Se esta também estiver obstruída, vou escolher aleatoriamente uma das direções que estejam disponíveis para seguir. Contudo, os fantasmas só fazem este "processamento" nas interseções do mapa, para evitar que fiquem presos num mesmo sítio quando o caminho ideal não está livre.
- **Fuga** – este modo fica ativo sempre que o Pac-man consome um energizer, desde que o fantasma não esteja a ser controlado por um utilizador. Nele, o fantasma usa um pensamento semelhante ao do modo de busca, mas em vez de seguir na direção em que se encontra o Pac-man, segue naquela que lhe é oposta, mas com probabilidade de erro.

Isto é, quando se encontra em fuga, o fantasma em 60% das vezes segue a direção oposta à do Pac-man, e nas restantes segue uma aleatória, para facilitar a sua captura pelo Pac-man. Se ele decidisse por seguir na direção oposta, no exemplo referido no modo de busca, a ordem de prioridade nas direções a seguir seria S, W seguidas de N ou E. Tal como no modo de busca, neste modo a decisão da direção a seguir pelo fantasma apenas é feita nas interseções do mapa.

Contudo, os 4 fantasmas têm métodos de funcionamento diferentes, constantes para cada fantasma e baseados nos 3 modos referidos acima. Esta funcionalidade também existia no jogo original, embora fosse implementada de uma forma ligeiramente diferente. No nosso caso, os fantasmas têm o seguinte comportamento:

- **Fantasma laranja** – este é o fantasma "idiota", pois está permanentemente em modo aleatório, exceto quando se encontra em modo de fuga (quando o Pac-man consome um energizer).
- **Restantes fantasmas** – estes fantasmas alternam constantemente entre o modo aleatório e de busca, estando algum tempo em cada um deles. Isto é, cada um é programado de forma a estar X segundos em modo de busca e Y segundos em modo aleatório. Assim, uma vez passados X segundos ele passa para o modo aleatório. Passados Y segundos neste modo, ele regressa ao modo de busca, e o ciclo repete-se.

## Bónus

Os bónus de pontuação surgem sobre a forma de frutas (laranja, cereja, morango, banana e pêra) e aparecem no local em que surge o Pac-man no início do jogo. Estes surgem apenas de X em X segundo e durante um curto período de tempo. De cada vez que surge um bónus, ele pode assumir a forma de qualquer uma das frutas (com probabilidade igual para cada uma). Quando o Pac-man consome um bónus enquanto este está visível, ele volta a deixar de estar disponível e atribui 200 pontos ao Pac-man (independentemente da fruta representada nesse momento).

# Estado do projeto

Todas as funcionalidades acima referidas foram implementadas com sucesso no Pac-nix. Gostaríamos de ter implementado também uma mudança na velocidade do Pac-man ou dos fantasmas quando o primeiro consome um energizer, mas tal não foi possível devido à forma como foram estruturadas algumas partes do programa na sua fase inicial. Quando nos apercebemos dessa "falha" (que pode também ser vista como uma característica do nosso jogo, isto é, por ser diferente do jogo original não significa que esteja errado), poderíamos corrigi-la mas isso implicava alterações em mecanismos base do jogo, pelo que optámos não o fazer para não arriscar estragos graves e difíceis de detetar no programa.

Também não implementámos a porta de série, apesar de estar referida na nossa proposta de projeto, porque nos foi impossível devido a falta de tempo e dificuldades na tentativa da sua programação. Tentamos implementá-la em programas externos, mas como não conseguimos não a chegamos a implementar no projeto final. Em fases finais do projeto pensámos por várias vezes incluí-la no projeto, mas focámo-nos em fazer um jogo muito estável e completo, pois preferíamos isso a ter um jogo instável, com erros e menos "perfeito" só para ter uma tentativa de porta de série. Como observação, pretendíamos usar a porta de série para permitir que o modo multiplayer funcionasse em computadores separados, em que um controlaria o Pac-man (este seria também o host da ligação) e o outro controlaria os fantasmas (sendo o client).

Outra questão que gostaríamos de ter implementado era um modo de vídeo diferente, que permitisse a utilização de bitmaps, mas quando nos apercebemos que podíamos alterar isso já tínhamos muitas imagens xpm feitas e tudo implementado em função do modo 0x105, pelo que decidimos não alterar isso pelas mesmas razões apresentadas em relação à velocidade do Pac-man e dos fantasmas.

Estes foram os periféricos que implementamos com sucesso:

| Periférico | Funcionalidade | Com interrupções? |
|---|---|---|
| **Timer** | Controlo dos frames e das animações das imagens, bem como controlo de ações dependentes de tempo (p.ex., alternação entre modos nos fantamas) | Sim |
| **Teclado** | Controlo no jogo e menus | Sim |
| **Rato** | Controlo no jogo (multiplayer) e menus | Sim |
| **Video card (modo gráfico)** | Visualização de menus, jogo e imagens | Não |
| **RTC** | Apresentação da data no menu principal | Não (por Polling) |

- **Timer** – O timer foi utilizado para o controlo do aparecimento dos frames no ecrã, bem como para gerir animações e outras situações dependentes do tempo, como a alternância entre modos nos fantasmas, o "piscar" dos energizers e dos fantasmas, o aparecimento e desaparecimento dos bónus, etc. Nenhuma destas funcionalidades implicou a alteração da frequência do timer, uma vez que isso implicaria também uma alteração na data do Minix o que não seria aconselhável. A maior parte das funcionalidades do timer estão implementadas nos ficheiros "timer.h" e "timer.c", bem como na função "fps_tick()" presente no ficheiro "pacnix.c".

- **Teclado** – o teclado foi utilizado para fazer a seleção das opções nos menus e para controlar Pac-man e fantasmas no jogo. Estas implementações estão disponíveis nos ficheiros “kbd_header.h”, “kbd_funct.h”, “kbd_funct.c” e mais especificamente nas funções “game_local()” e “start_menu()” do ficheiro “pacnix.c”. Exitem outras (por exemplo no ficheiroc “pac_menu.c”), mas não são tão fundamentais no programa.

- **Rato** – o rato é utilizado para selecionar as opções do menu e para selecionar o fantasma a controlar no modo multiplayer. A implementação do rato está nos ficheiros “mouse_header.h”, “mouse.c” e “mouse.h”, e também mais especificamente nas funções “game_local()” e “start_menu()” do ficheiro “pacnix.c”.

- **Video card** – a placa de vídeo é utilizada em modo gráfico (modo 0x105) para apresentar toda a informação gráfica do jogo, desde menus ao jogo em si. A maior parte da implementação foi feita nos ficheiros “video_gr.h”, “video_gr.c”, “videobe.c”, “videobe.h” e “video.h”, embora alguns detalhes surjam nos ficheiros “texto_num.c”, “pac_menu.c” e “maze.c”.

- **RTC** – o rtc (Real Time Clock) é utilizado para mostrar no menu do jogo, a data e a hora. A implementação do rtc está nos ficheiros “rtc.c” e no “rtc.h”.

## Organização de código

Na realização do projeto tentámos separar bem os módulos existentes, de forma a ter módulos base (por exemplo, aqueles que tratam informações específicas dos periféricos), para poder depois contruir módulos mais complexos que os utilizassem, por exemplo, o módulo principal do jogo (“pacnix.c”). Segue então a descrição de cada módulo:

### Módulo “asprite.c”

Este módulo possui a informação relativa a sprites animadas do nosso programa, isto é, sprites que necessitem de transitar entre várias imagens diferentes de forma a criar uma animação fluida.

O módulo foi desenvolvido em conjunto e igualmente por ambos os membros do grupo (50/50), tendo cada um contribuído aproximadamente o mesmo para a sua realização. Consideramos que este tem um peso de 4% no trabalho, uma vez que é usado para a animação do Pac-man.

## Módulo “kbd_funct.c”

Este módulo possui informação relativa à interação com o teclado. As principais funcionalidades por ele implementadas são a subscrição e cancelamento das interrupções do teclado, bem como a leitura de scan-codes das teclas do teclado, importantes para a utilização do teclado como mecanismo de controlo nos menus e no jogo.

Este módulo foi também desenvolvido igualmente (50/50) pelos membros do grupo, essencialmente nas aulas laboratoriais. Consideramos que tem um peso de 2% no trabalho, uma vez que, apesar de ser imporante, é bastante simples e já estava praticamente feito no lab das aulas práticas.

## Módulo “main.c”

Este é provavelmente o módulo mais simples do projeto, porque simplesmente inicializa o modo gráfico da placa de vídeo e chama a função que inicializa os menus do jogo. O módulo foi também desenvolvido em conjunto por ambos os membros do grupo (50/50), representando apenas 1% do projeto final devido à sua simplicidade.

## Módulo “maze.c”

Este é também um módulo muito importante para o projeto, uma vez que trata da apresentação do mapa do jogo, bem como de colocar e retirar pac-dots e energizers à medida que são consumidos. Também permite a implementação do teletransporte do Pac-man nos pontos respetivos.

A implementação do mapa do jogo foi baseada num sistema de peças semelhante a um puzzle. Na verdade, o mapa é constituído por vários quadrados com 30 pixéis de lado, com desenhos diferentes que se unem entre si, e que, em conjunto, desenham o mapa do jogo. Existe, portanto, um array de inteiros, que representa o número das peças a colocar em cada zona do mapa.

Para além disso, este módulo visa a simplificação da criação de novos mapas de jogo, pelo que, se fosse desejado, seria bastante simples criar um ou vários novos mapas, bastando simplesmente dispor de forma diferente as peças do mapa presentes no ficheiro “maze.h”.

A parte gráfica deste módulo, isto é, as imagens das várias peças do mapa foi quase totalmente feita pelo Leonardo, enquanto a parte do código foi feita em conjunto pelos dois membros do grupo, embora talvez um pouco mais pelo André. Pode-se considerar que o Leonardo fez 60% deste módulo e o André 40%. Consideramos também que este módulo representa 13% do peso do projeto, pois foi essencial para a apresentação do mapa do jogo.

## Módulo “mouse.c”

Este módulo implementa uma grande parte das funcionalidades do rato, incluindo operações com interrupções, leitura de packets, ativação ou inibição do modo stream. Para além disso, possui abstrações para duas “classes”, Mouse_coord e Mouse_packet, que representam, respetivamente, o cursor do rato e um packet lido aquando das interrupções. Estas “classes” facilitaram imenso a coordenação do rato e das suas interrupções no projeto, simplificando a estrutura do rato e o acesso às informações deste.

Este módulo foi feito parcialmente para o lab das aulas laboratoriais, sendo que a outra parte foi feita mais tarde, no início do projeto. Consideramos que foi feito 60% pelo André e 40% pelo Leonardo e que tem um peso de 6% no projeto, pois facilitou a utilização do rato em todas as partes do jogo.

## Módulo “pac_menu.c”

Este módulo implementa a apresentação dos menus do jogo, bem como as suas interações com o rato e o teclado. Foi um dos últimos a ser desenvolvido, e mostrou-se bastante trabalhoso pela transição entre menus e pela complexidade das imagens que são exibidas, bem como pelo seu tamanho e características de cor.

Neste caso, todas as imagens foram feitas pelo Leonardo, enquanto a parte do código foi feita por ambos, embora um pouco mais pelo André. Consideramos que a contribuição de ambos, apesar de ter um teor diferente, foi igual no que toca ao trabalho dispensado para ambas, sendo de 50% para cada um. Consideramos também que o módulo tem um peso de 6% no projeto.

## Módulo “pacnix.c”

Este é o módulo fundamental do jogo. É ele que implementa a maior parte das funcionalidades referentes ao jogo em si. Desde movimento dos personagens, a interação do utilizador nos menus, ou deteção de colisões, ou tratamento de exceções, tudo isso é feito neste módulo. É também ele que possui as abstrações a “classes” Pac-man, Ghost e Bonus que são essenciais para o funcionamento do jogo.

É também neste módulo que se faz a ligação de todos os periféricos que usamos, e no qual é percetível o contributo de cada um deles para que as mecânicas do jogo funcionassem.

Consideramos que este módulo tem um peso de 48% no jogo. Embora ambos tenham trabalhado para este módulo, uma grande parte deste foi feito pelo André, embora o Leonardo tenha ajudado e contribuído em várias partes. Consideramos que o André fez 80% deste módulo, e o Leonardo 20%.

## Módulo "read_xpm.c"

Este módulo foi-nos fornecido pelos professores para as aulas laboratoriais, e aplicamo-lo ao nosso projeto. Ele é usado sempre que é necessário "ler" uma imagem em formato xpm de forma a usá-la no programa.

Apesar de ser muito utilizado no programa, este é um módulo muito simples e que não foi desenvolvido por nós, por isso consideramos que apenas representa 2% do projeto final.

## Módulo "sprite.c"

Este foi também um dos módulos mais usados no programa, pois contém a abstração para a "classe" Sprite, que representa toda a informação importante para uma imagem estática (sem animação, mas que pode ter movimento).

Mais uma vez, como também é um módulo simples, consideramos que apenas representa 1% do projeto final, tendo sido feito em igualdade (50/50) por ambos os membros do grupo.

## Módulo "texto_num.c"

Este módulo é utilizado para a escrita de números inteiros no ecrã, sendo neste caso utilizado para mostrar a pontuação do jogador. Apesar de ser um módulo importante no projeto acaba por ser pouco utilizado, porque o seu impacto no jogo em si é pouco, apenas o de permitir a visualização da pontuação.

O módulo foi desenvolvido totalmente pelo André (100%), representando 5% do projeto final.

## Módulo "timer.c"

Este é também um módulo bastante simples que trata tudo que diga respeito ao timer, desde a sua configuração, a subscrição de interrupções e algumas outras funcionalidades. Ele foi desenvolvido em conjunto e igualdade (50/50) pelos membros do grupo, representado 2% do trabalho final.

## Módulo "video_gr.c"

Este é um módulo de complexidade intermédia mas de alguma importância para o projeto, uma vez que engloba tudo que trate o aparecimento de imagens no ecrã, bem como a configuração do modo de vídeo, o tratamento de cores "transparentes", etc. Ele foi

desenvolvido tanto nas aulas laboratoriais como posteriormente de forma específica para o projeto, representando cerca de 5% do trabalho final, e tendo sido feito 65% pelo André e 35% pelo Leonardo.

## Módulo "videobe.c"

Este módulo foi desenvolvido unicamente para as aulas laboratorias referentes à placa de vídeo, apenas tratando da configuração desta. Consideramos que foi desenvolvido igualmente por ambos os membros do grupo (50/50) e que tem um peso de 1% no trabalho final.

## Módulo "vmem_txt.c"

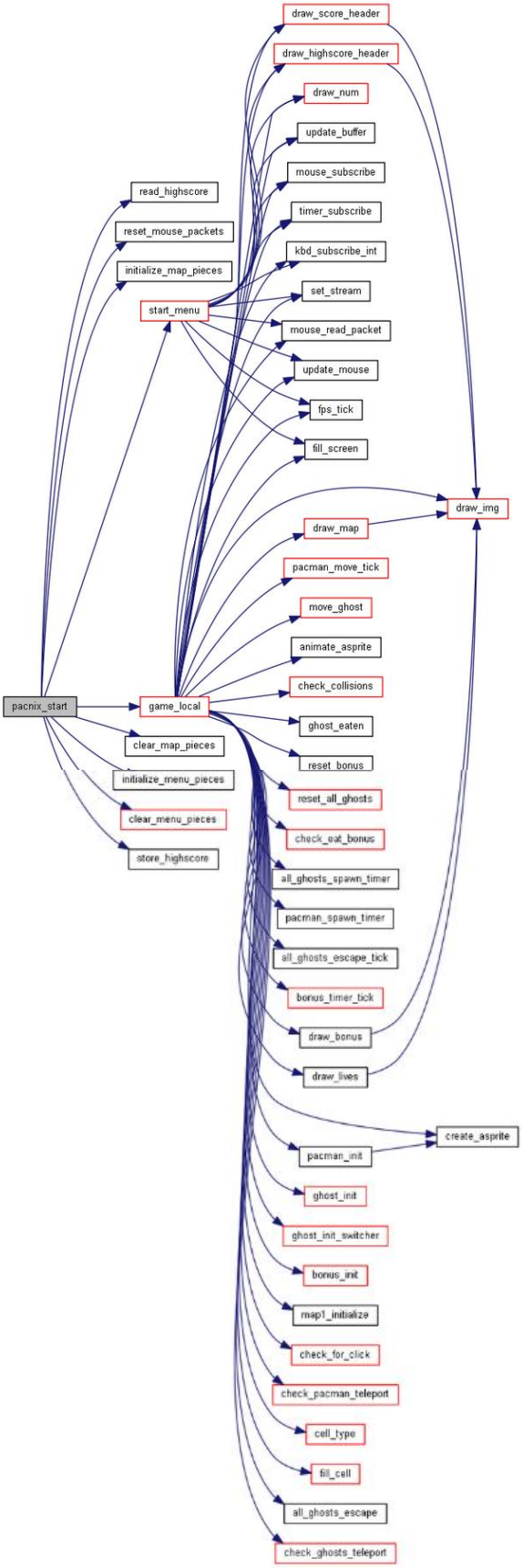
Este módulo foi desenvolvido unicamente nas aulas laboratoriais de LCOM, e acabou por não ser usado no projeto uma vez que é referente a propriedades da placa de vídeo em modo de texto, modo que não foi utilizado no projeto. Foi desenvolvido igualmente por ambos os membros do trabalho mas representa 0% do projeto final.

## Módulo "rtc.c"

Este módulo foi construído com base no lab6 do ano passado e desenvolvido para o jogo "seguir" a data e a hora do PC mesmo quando este está desligado. Não apresenta grande complexidade uma vez que a maior parte das operações representam acessos a registos e a prints de informação de cada desses mesmos. O módulo foi utilizado para apresentar a data e hora atual no menu do jogo.

O módulo foi desenvolvido totalmente pelo Leonardo (100%), representando 4% do projeto final.

# Diagrama de Chamada de Funções



## Descrição das principais funções

Antes de descrever as principais funções do programa, vamos fazer uma pequena referência à função "main()". Essa função não foi referida no gráfico da página anterior, uma vez que apenas faz uma inicialização geral de alguns aspetos do ambiente de jogo, nomeadamente o modo gráfico (com vg_init() e vg_exit()), a geração da seed para o random (srand()) e a atribuição das operações iop para o programa (sys_enable_iop()). De seguida, a função apenas chama pacnix_start(), que dá início ao jogo.

As principais funções do programa, que gerem as funcionalidades principais do jogo e que fazem interação com as interrupções dos periféricos (através da função driver_receive()) são as funções "game_local()" e "start_menu()".

- **"game_local()"** – esta função representa o motor do jogo em si, sendo polivalente pois permite concentrar nela os modos singleplayer e multiplayer. É nesta função que se faz a subscrição das interrupções necessárias ao funcionamento do jogo, bem como a inicialização de variáveis que guardam informações importantes da jogabilidade (por exemplo, o pacman, os fantasmas, os bónus, a pontuação, entre outras). A função, após inicializar o mapa e aspeto iniciais do jogo, deteta o aparecimento de interrupções, chamando as funções necessárias à interpretação (handling) destas para o jogo (por exemplo, mover o pacman quando se prime uma seta do teclado).
- **"start_meu()"** – esta função trata da apresentação dos menus do jogo, bem como da interação entre estes e o jogo (no sentido em que indica à função que a chama o modo de jogo adequado a iniciar). É, assim, uma função polivalente, pois conjuga a capacidade de mostrar três menus diferentes (menu principal, menu de instruções e menu de "créditos"). Para fazer isto, a função concilia as funcionalidades de vários periféricos (placa de vídeo, rato, teclado, RTC e temporizador) através da função "driver_receive()", para uma série de funcionalidades, como movimento do cursor do rato, seleção das opções com rato ou teclado, mostra da data e da hora atuais, entre outras.

# Detalhes da implementação

## Aspetos a salientar

Em relação aos aspetos do trabalho que foram lecionados na unidade curricular, consideramos que alguns apresentaram maior complexidade e dificuldade do que estávamos à espera. Por exemplo, no que toca a ter um programa que possa possuir vários estados entre os quais transita, foi um pouco complicado gerir a transição, por exemplo, entre os menus do jogo e o jogo em si, principalmente pela necessidade de subscrever ou impedir interrupções sempre que se entra e sai de cada um desses estados, principalmente em periféricos como o rato, no qual por vezes os primeiros packets são difíceis de ler ou trazem informações incorretas. Também, inicialmente, a contagem/geração do número pretendido de frames por segundo não

funcionou corretamente, mas esta questão foi posteriormente resolvida com relativa facilidade, e recurso a alguma matemática.

Em relação aos aspetos não mencionados nas aulas, alguns revelaram-se especialmente complicados e difíceis de implementar. Destes, salientamos três: movimento de personagens, deteção de colisões e inteligência artificial dos fantasmas. Nas fases iniciais do projeto, implementar a movimentação de personagens que não atravessassem paredes e fossem capazes de seguir por caminhos predefinidos e "estreitos" foi bastante complexa, devido à velocidade de movimentação dos personagens, bem como evitar que o utilizador tivesse de acertar no momento certo para premir uma tecla, quando quisesse que o personagem virasse, para seguir um corredor. Se o personagem se virasse no momento em que o utilizador prime a tecla, muitas vezes ficaria parado, virado contra a parede, a uns pixéis do corredor. Contudo, com o sistema que implementamos, o personagem, após ter sido premida a tecla, apenas vira quando tiver um caminho livre para seguir. Assim, o utilizador pode premir a tecla um pouco antes de chegar a uma interseção que o personagem irá saber quando virar.

Em relação à deteção de colisões, também sentimos alguma dificuldade, pela necessidade de detetar colisões com várias entidades (paredes, fantasmas, portais, pac-dots, energizers e bónus). Contudo, conseguimos resolver todas estas colisões com recurso a técnicas diferentes (por exemplo, medida da distância entre o pacman e os fantasmas, ou pela verificação sobre em que célula do mapa se encontra o pacman).

Por fim, talvez um dos aspetos mais complexos do nosso programa foi a inteligência artificial dos fantasmas. Fazer fantasmas que se movimentassem aleatoriamente pelo mapa teria sido relativamente fácil. Fazer fantasmas que seguem o caminho mais próximo para o pacman, ou que alternam entre segui-lo ou andarem aleatoriamente, foi bastante complicado, devido a vários factores, como a existência de obstáculos, e, por vezes, a impossibilidade de seguir o caminho ideal para o pacman. Implementações iniciais faziam com que os fantasmas ficassem "indecisos", ficando parados num mesmo sítio, deslocando-se uns pixéis para um lado, voltando atrás logo de seguida, e repetindo o mesmo comportamento enquanto o pacman não mudasse de local. Resolver conflitos deste género foi muito complexo, não só a nível da programação em si, mas também dos algoritmos por trás da deteção do caminho mais curto a seguir. Existiam várias formas possíveis de resolver este problema, soluções essas que se baseavam em técnicas diferentes de path-finding, mas utilizamos a que descrevemos anteriormente no relatório, por ser aquela que, para nós, apresentava uma maior relação eficácia/(complexidade + tempo de execução).

# Últimas considerações

## Avaliação do curso:

Com base em todo o trabalho realizado e com todo o conhecimento que conseguimos adquirir, podemos afirmar que esta cadeira foi bastante produtiva e esclarecedora quanto aquilo que queremos seguir no futuro mais em especifico. Diríamos até que nos permitiu evoluir em muitos aspetos desde explorar os vários periféricos como o rato, o teclado e placa de vídeo até criar um projeto onde todos os periféricos anteriormente mencionados e mais alguns podem interagir e resultar em algo que nos deixa com um sentimento de satisfação e de entusiamo.

Para além disso, achamos bastante positivo o facto de nos ser possibilitada a ajuda por parte dos monitores (no nosso caso Henrique Ferrolho) pois sem eles seria difícil superar algumas das adversidades que encontramos durante o nosso trajeto.

Também achamos positiva a estruturação da unidade curricular, no sentido em que foi feito um avanço gradual pelos periféricos, bem como o facto de a avaliação distribuída ser feita sobre estes, gradualmente.

Como aspetos negativos, diríamos que a calendarização dos labs e as datas de entrega dos mesmos não foi muito eficaz (visto que algumas turmas tinham de fazer os labs antes de assistir às aulas teóricas todas sobre este; por vezes, algumas dificuldades que nos surgiam eram esclarecidas nas teóricas já depois do nosso prazo de entrega, mas antes do prazo de outras turams). Para além disso, a falta de aulas práticas relativas ao desenvolvimento do RTC e da porta de série prejudicou-nos no sentido em que tornou mais difícil a implementação destes periféricos no projeto. Para aqueles que queriam implementar o RTC e a porta de série no trabalho, como era o nosso caso, era essencial termos alguma prática e ajuda nos labs de RTC e porta de série, o que não foi possível. Quanto ao tempo de avaliação, sentimo-nos algo limitados no tempo uma vez que em alguns casos foi bastante difícil entregar os labs a tempo.

## Auto-avaliação:

| | |
|---|---|
| Leonardo: | André: |
| Participação: 45% | Participação: 55% |
| Contribuição: 40% | Contribuição: 60% |

## Instruções de instalação:

Antes de se de avançar, deve-se garantir que todas as operações que envolvam jogo são feitas com acesso root ao sistema. Para que tal aconteça, antes de partir para a instalação e execução do programa, deve-se utilizar o comando “su” para garantir todas as permissões necessárias ao programa.

- ➔ Para instalar o jogo, deve-se navegar para a pasta “proj” do projeto, e executar os scripts de instalação e compilação (seguindo esta ordem), usando os comandos “sh install.sh” e “sh compile.sh”.
- ➔ Para iniciar o jogo, basta, na mesma pasta, executar o script de execução, com o comando “sh run.sh”.