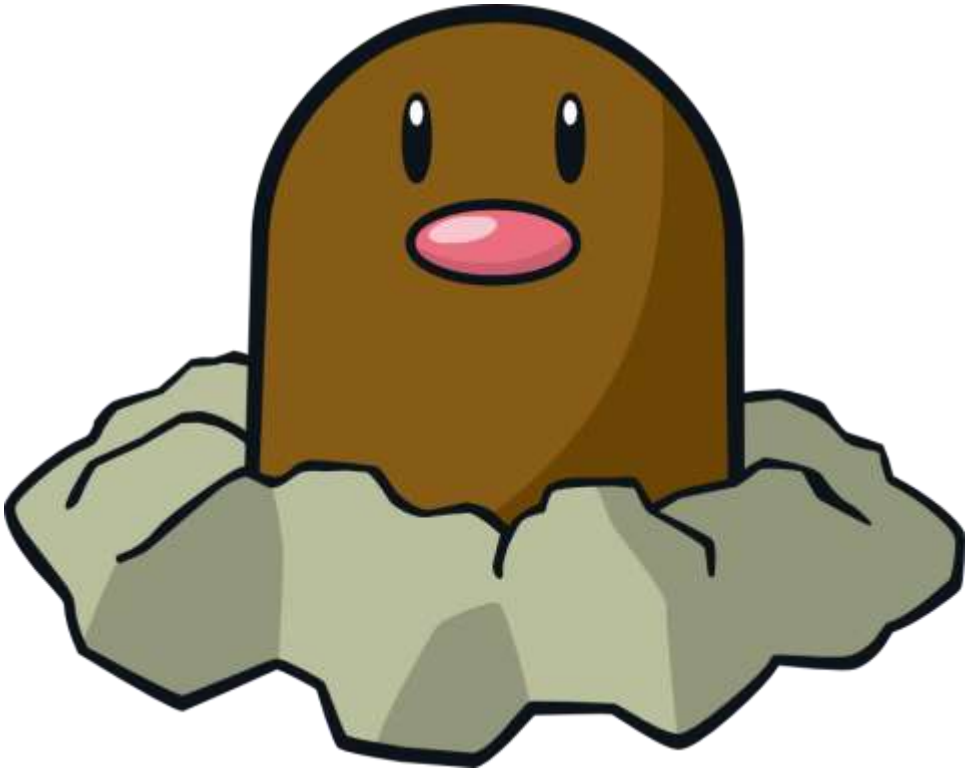
Laboratórios de Computadores:

PROJECTO - RELATÓRIO FINAL
WHACK-A-MOLE
LCOM1415-T3G15



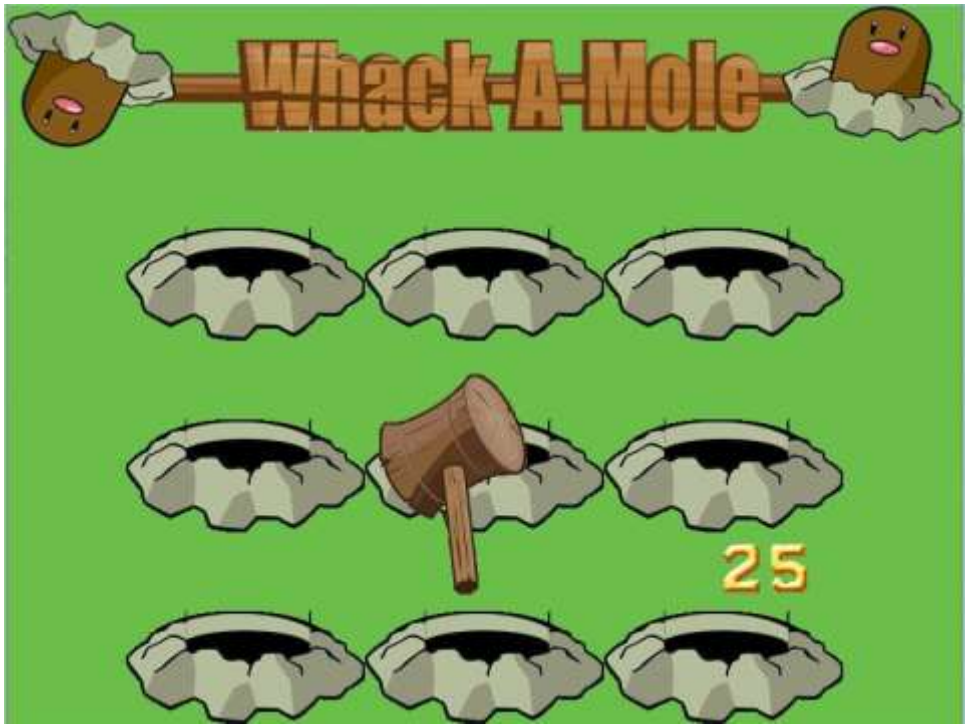
João António Custódio Soares ei12093
Mário Filipe Araújo Ferreira ei12049

Tabela de Conteúdos / Índice:

Sobre o jogo Whack-A-Mole:

Whack-A-Mole é um jogo clássico inventado em 1976 e jogado tipicamente numa máquina do tipo Arcade que contem 5 buracos no topo. Destes buracos surgem uns bonecos parecidos com umas marmotas (Mole) que estão constante movimento para cima e para baixo sendo que quando estão em baixo estas se encontram escondidas dentro dos buracos. O objetivo do jogo é forçar cada um dos bonetos individualmente de novo para dentro dos seus buracos usando para isso um martelo de plástico geralmente cilíndrico. O Jogador deve usar este martelo para bater na cabeça do boneco ganhando assim um ponto. No final do jogo são contados os pontos e quanto mais rápido se pontoar, mais pontos se faz.

Na tentativa de recriar este jogo, resolvemos fazer uma adaptação em que a lógica do jogo (pontuar ao acertar num boneco) se mantém e associamos os movimentos do martelo ao movimento e cliques no rato. As moles vão aparecer de forma aleatória no jogo, e para pontuar, o jogador deve mover o rato até um boneco que esteja fora do seu buraco e fazer um clique de rato para simular o bater do martelo, sendo que se este falhar, poderá tentar de novo até o tempo de jogo acabar.



Utilização de periféricos:

Para o desenvolvimento deste projeto utilizamos os seguintes periféricos:

Teclado: Utilizamos o teclado configurado através de interrupções. Este terá como função iniciar o jogo a partir do menu principal, servindo-se das setas para navegar e selecionar as opções desejadas. O jogador poderá também premir a tecla ESC para interromper o jogo para Pausa e terminá-lo.

Rato: O periférico do rato será também configurado com interrupts, sendo a sua principal função simular o movimento do martelo e a "martelada" durante o jogo. Também poderá ser usado para navegar pelos menus inicial e de opções.

Placa de Vídeo em modo gráfico: será utilizada para mostrar ao jogador todo o conteúdo gráfico do jogo.

Temporizador : O periférico do temporizador será utilizado como um contador e funcionará também com interrupts. Este contador é decrementado todos os segundos, e se chegar a 0, o jogo termina.

Instruções de Utilização:

Implementamos uma UI gráfica recorrendo ao uso de bitmaps com imagens que constituíam as diferentes opções dos menus e às coordenadas dos cliques do rato para input. O rato, durante toda a execução, é também representado por uma imagem bitmap (uma marreta) que se altera aquando um clique do botão esquerdo para simular a "marretada".

Main Menu:

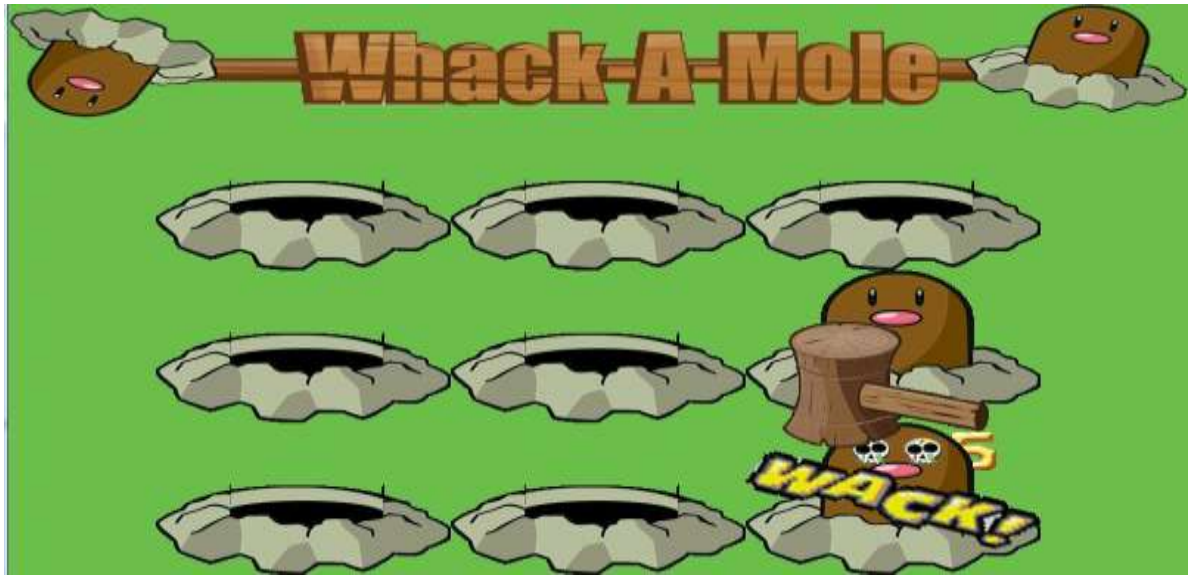
A partir deste estado pode selecionar iniciar um jogo, listar as melhores pontuações, ou aceder ao menu das opções. Pode também optar por terminar a execução.

Menu das Opções:

A partir deste estado pode aumentar a dificuldade e regular o tempo de jogo. A dificuldade define a frequência com que as marmotas emergem dos buracos, e o tempo de jogo regula o temporizador (timer) que dita o fim do jogo.

Jogo:

Este é o estado de jogo, em que utilizamos uma imagem bitmap para cada estado das moles. Encontram-se centrados neste 9 posições que representam as várias "casas" para as moles. Quando as moles não se encontram disponíveis para serem atingidas, utilizamos uma imagem de um buraco apenas. Quando estas se encontram visíveis, utilizamos uma imagem de um pokémon (Diglett) que, quando deteta um clique do rato se altera para uma imagem desse mesmo pokemon, só que com a palavra "whack" (daí o jogo se chamar Whack-a-Mole). Cada "whack" bem concretizado confere ao jogador um ponto que incrementa a sua pontuação que também é apresentada neste estado. Quando termina o tempo de jogo, a pontuação é registada para Highscores e o jogador é remetido ao menu inicial.



Estado do Projecto:

Periférico	Função	Interrupções	Ficheiros .c .h
KBD	navegação nos menus	Sim	keyboard
Mouse	navegação nos menus e input no jogo.	Sim	Mouse
Timer	Contabilização do tempo de Jogo	Sim	timer
Video Card	GUI	Não	Graphics

O objetivo principal deste trabalho foi concretizado com sucesso, sendo que conseguimos configurar os periféricos necessários para garantir uma lógica de jogo funcional, sendo que não conseguimos implementar todos os nossos objetivos da especificação do projeto. Não tivemos tempo para completar a configuração do periférico RTC para gravar a data de uma pontuação após um jogo. Deixamos este periférico de parte pois decidimos focar-nos nos mais importante para garantir navegação pelo projeto e uma execução fluída do jogo.

Arquitetura e Organização do Código:

Este projeto está estruturado em módulos. A cada um desses módulos estão associados ficheiros presentes na pasta code.

Módulo Principal:

main.c - Inicialização do modo gráfico no modo 0x114 (800x600pix), criação do gerador de números aleatórios para ser utilizado na lógica de jogo e ciclo de controle (lifecycle) do jogo.

Módulo UI:

MainMenu(.c e .h) - Menu principal e mostrado inicialmente onde se pode escolher as próximas opções de navegação da UI.
OptionsMenu(.c e .h) - Menu das opções onde se pode escolher o tempo do temporizador de jogo e a dificuldade.

Módulo Input:

keyboard(.c e .h) - Subscrição e desativação das interrupções do teclado.
Mouse(.c e .h) - Utiliza uma classe de C para representar a abstração do rato, guardando as informações relativas ao tratamento do input. Contém a função mouseInsideRect que determina se o rato se encontra numas determinadas coordenadas e getMouse para aceder à classe.

Módulo Gráfico:

VBE(.h e .c) - Para utilizar a parte gráfica em modo de vídeo.
Graphics(.h e .c) - Funcionalidades de uma biblioteca gráfica
Bitmap(.h e .c) - Foi criada uma classe para carregar, visualizar e destruir bitmaps.

Módulo Temporizador:

timer(.h e .c) - Contabilização do tempo de jogo definido nas opções.

Módulo Utilitário:

Utilities(.h e .c) - Seria utilizado para funções relacionadas com read e write para ficheiros para gravar as pontuações.
Numbers(.h e .c) - Função drawNumber para mostrar a pontuação e o tempo de jogo ao utilizador a partir de imagens bitmap de numeros.

Módulo da Lógica de Jogo:

Whack(.h e .c) - Representa a máquina de estados e serve-se das funções create, update draw e changeState para controlar a execução do programa. Tem três estados possíveis (MainMenu, Options e Game) e gere o término do jogo através da função isDone.
Game(.h e .c) - Classe do estado de jogo, utiliza as funções mouseInsideHoleNum, updateHoles, resetHoles e isFinished para representar as moles a entrarem e saírem dos

seus buracos (recorrendo a um RNG (random number generator)) e para contabilizar a pontuação do jogador.

Detalhes de Implementação:

Neste trabalho utilizamos uma máquina de estados para gerir a lógica de jogo. Esta máquina de estados processa as interrupções dos periféricos e é inicializada no inicio da execução do programa. Após a inicialização, esta consegue processar os eventos dos diferentes estados e o input do utilizador. Inicialmente é criada no estado de MAINMENU e cria o novo estado e apaga o anterior aquando de uma mudança de estado, guardando e passando as informações necessárias para o estado seguinte. No fim da execução do programa esta máquina de estados é apagada e termina-se o processo.

O ciclo de jogo é simples, as coordenadas dos buracos para as moles são guardadas num array e utilizadas para as verificações do input do rato e para o desenho das BMPs. Cada buraco é representado por um elemento do array holes que guarda o estado de cada mole. As moles podem-se encontrar no estado 0 (mole dentro do buraco, não disponivel para ser atingida), 1 (mole fora do buraco mas ainda não atingida) ou 2 (mole atingida por uma "martelada") que faz variar a imagem utilizada para a representar. Cada "martelada" numa mole no estado 2 confere um ponto ao jogador e o jogo termina quando o timer chegar ao limite definido (inicializado como 30 segundos). O estado de cada mole é atualizado por um RNG que define o seu estado como 0 ou 1 a cada iteração.

Instruções de Execução:

#service run /home/lcom/lcom1415-t3g15/proj/code/projeto

Discussão:

Achamos que a nossa resolução do trabalho foi positiva apesar de não termos completado todas as metas propostas na entrega intermédia, nomeadamente, não implementamos o sistema de highscores nem incluímos a utilização do periférico RTC. Deixamos estes pontos para último já no nosso planeamento pois eram os que tinham menos importância para a boa execução do programa e não conseguimos completar a tempo as funcionalidades que a permitissem.

Conclusão:

Apesar das falhas nas nossas metas estabelecidas, acho que aprendemos muito com este trabalho especialmente devido à boa coordenação e sinergia que encontramos um com o outro ao longo do desenvolvimento deste. Esta dinâmica entre nós mostrou-se muito importante porque facilitou muito o trabalho de grupo e garantiu que estávamos sempre os dois a par do desenvolvimento deste. Durante as aulas laboratoriais já tínhamos este projeto em mente, e esse planeamento fez com que a divisão de tarefas fosse eficaz e precisa para ambos percebermos e avançarmos no desenvolvimento com celeridade. No geral adquirimos os conhecimentos necessários e podemos dizer que o método de ensino e planeamento das aulas foi mais pedagógico este ano comparado com o ano passado, o que garantiu que completássemos todos os objetivos propostos este ano.

Autoavaliação:

Como já foi referido, conseguimos conciliar as tarefas de modo a que ambos os elementos do grupo conseguissem estar a par de tudo, e foi um trabalho de equipa e programação em pares em que dividimos a responsabilidade dos módulos pelos elementos da seguinte forma:
João Soares : Módulos UI, Gráfico e Input;
Mário Ferreira: Módulos da Lógica de Jogo e temporizador;
Sendo que o Módulo de Input em particular foi desenvolvido puramente por peer-programming (duas pessoas a programar na mesma máquina) e achamos que no final das contas ambos dispusemos o mesmo trabalho e o esforço para concretizar este trabalho com sucesso.