Grupo 20

Clara Pais, nº 92622

Sara Marques, nº93342

Licenciatura em Engenharia Informática e de Computadores
2020/2021

Projeto de Introdução à Arquitetura de Computadores

De seguida encontra-se um pequeno relatório do programa em *Assembly* do P4 que implementa uma versão simplificada do jogo Dino. É apresentada uma breve descrição de todas as constantes e variáveis globais usadas e sub-rotinas implementadas. Na nossa implementação valorizamos aspetos como a representação do dinossauro que em vez de ser só um caracter, é um desenho mais elaborado. Isto pode trazer a desvantagem de atrasar o processo de atualizar o jogo por ter funções mais demoradas e não ser tão fácil (ou mesmo impossível) passar em algumas sequências de cactos (isto também se deve ao algoritmo de geração de cactos fornecido). No entanto, preferimos optar por este método pois torna o jogo mais interessante e traz outros desafio na sua implementação. Para facilitar a jogabilidade e apesar de não ser pedido no enunciado, implementamos o funcionamento do salto clicando no espaço do teclado (para além da seta para cima. Acrescentámos também o começar o jogo clicando no '0' do teclado (para além do botão 0) também para melhorar a experiência do jogo.

Constantes

Para além das constantes usadas na primeira entrega do projeto, usámos as constantes do P4 relacionadas com a janela de texto, o *display* de 7 segmentos, o *timer* e a máscara de interrupções. Declarou-se constantes relacionadas com as posições no terminal por exemplo a posição do terreno, das nuvens, do dinossauro inicialmente, de mensagens de texto, altura máxima do salto do dinossauro…Estão também declaradas as constantes relacionadas com as cores utilizadas na representação do jogo.

Variáveis globais

Foram utilizadas variáveis auxiliares para o funcionamento do *timer* e do *display* de 7 segmentos. Declarou-se também um conjunto de variáveis que descreve o estado do jogo em cada instante como a tabela JOGO que contem a posição dos cactos, a posição atual do dinossauro, assim como os seus extremos para facilitar a deteção de colisões. Usaram-se também várias *flags* para indicar por exemplo, se o jogo já começou, se é o primeiro jogo, se o dinossauro está a saltar, a direção do salto… Por último, declararam-se várias cadeias de caracteres para facilitar a escrita e desenho do jogo no terminal.

Sub-rotinas

principal: função principal do programa que escreve no terminal o ambiente de jogo (terreno, nuvens, dinossauro e mensagem inicial) antes deste começar. Configura a máscara de interrupções e permite a sua execução. Espera pela interrupção que dá inicio ao jogo. Existe também uma sub-rotina **PROCESS_CHAR** neste ciclo para detetar quando é pressionado o '0' do teclado para dar inicio ao jogo. Quando começa o jogo, permite também a interrupção da seta para cima que antes estava desativada. Se não for o primeiro jogo limpa tudo o que estava no terminal. Caso contrário apaga só a mensagem inicial. Inicializa e começa a contar o *timer*. Entra num ciclo em que vai atualizando o *timer* e o jogo sempre que TIMER_TICK for diferente de 0. Neste ciclo, está também outra sub-rotina **PROCESS_CHAR** para detetar quando é pressionado o espaço que desencadeia o salto.

PROCESS_TIMER_EVENT: processa eventos do *timer*, diminui a variável TIMER_TICK e chama a função **conta** que atualiza a pontuação.

PROCESS_CHAR: lê o caracter pressionado e caso seja um '0', começa o jogo (muda a variável INICIA_JOGO para 1) ou um espaço, inicia um salto (coloca a variável SALTO a 1).

conta: atualiza o valor da pontuação, somando 1 ao valor atual.

WRITE_ON_DISP7: soma 1 ao valor mostrado no display, atualizando-o. Para isso, usa a função auxiliar **proximodigito** que determina o próximo *display* a ser alterado.

atualizajogo: função já explicada no relatório anterior com a adição de chamar as funções para atualizar a posição de um cacto no terminal ou simplesmente o imprimir (na ultima posição). Chama também a função que atualiza a posição do dinossauro se houver um salto a decorrer.

imprimeterreno: imprime as duas linhas que constituem o terreno de jogo a amarelo.

imprimenuvens: imprime cada linha que forma o desenho das nuvens a azul na parte superior do terminal.

imprimedino: imprime cada linha que forma o desenho do dinossauro a roxo numa determinada posição do terminal, passada como argumento em R1. O desenho pode variar dependendo se o jogo acabou ou não.

apagadino: apaga o dinossauro do terminal recebendo a sua posição como argumento (R1).

atualizasalto: atualiza o salto do dinossauro a uma velocidade constante acedendo à variável que guarda a sua posição a cada instante (POS_DINO_ATUAL). Começa por apagar o dinossauro da posição anterior. Depois vê o valor da variável DIRECAO_SALTO para saber se o dinossauro está a subir (nesse caso muda a sua posição 5 linhas para cima) ou a descer (muda a posição 5 linhas para baixo) guardando o valor final em POS_DINO_ATUAL. A variável EXTREMOS_DINO é também atualizada da mesma forma. Compara também a posição atual com a altura máxima do salto e se for igual muda a sua direção e com a posição inicial e se for igual o salto chegou ao fim do salto.

imprimecacto: imprime um cacto a verde no terminal, recebendo a sua posição e altura como argumento (R1 e R2 respetivamente). Esta impressão é feita caracter a caracter e antes de imprimir cada verifica se não colide com nenhum dos extremos do dinossauro. Se esse for o caso salta para a sub-rotina **gameover**. Se não, imprime o caracter.

apagacacto: apaga um cacto do terminal, recebendo a sua posição e altura como argumento (R1 e R2 respetivamente). Vai apagando carácter a caracter. No final como o cacto está na mesma linha que o solo e se apaga esse caracter tem de se voltar a repor a parte do terreno que foi apagada, imprimindo-o de novo.

atualizacacto: atualiza a posição de um cacto, recebendo a sua posição e altura como argumento (R1 e R2 respetivamente). Para isto, imprime o cacto na nova posição (se esta não for a primeira coluna do terminal) e apaga-o da posição anterior.

imprimegameover: imprime cada linha que forma as palavras ”game over” no terminal.

gameover: termina o jogo. Começa por configurar a máscara de interrupções para o seu valor original. De seguida, imprime o dinossauro com uma pequena animação no olho quando se perde e chama **imprimegameover**. Coloca a 1 a variável PRIMEIRO_JOGO e a 0 a variável INICIA_JOGO. Por último, salta para a etiqueta **espera** na função **principal** aguardando pelo inicio do próximo jogo.

limpatudo: limpa tudo o que estiver em memória, no terminal e no *display* resultantes do jogo antigo, voltando aos seus valores iniciais.

imprimelinha, apagalinha, imprimecaracter, apagacaracter: funções auxiliares para ajudar a escrita no terminal.

Rotinas das interrupções

TIMER_ISR : função que trata das interrupções do timer. Chama **AUX_TIMER_ISR** que é uma função auxiliar que aumenta a variável TIMER_TICK e prepara o timer para continuar a contagem.

KEYZERO: caso seja pressionado o botão 0 coloca a 1 a variável COMECA_JOGO e dá-se inicio ao jogo.

KEYUP: caso seja pressionada a seta para cima coloca a 1 a variável SALTO e inicia-se um salto do dinossauro.