LICENCIATURA EM ENGENHARIA INFORMÁTICA E DE COMPUTADORES

Projeto de
Introdução à Arquitetura de Computadores

Pássaro Bamboleante

2015 / 2016

INSTITUTO SUPERIOR TÉCNICO

Índice

1. Objetivo

O projeto consiste no desenvolvimento de um jogo em que o objectivo é um pássaro (que se movimenta de acordo com leis da física) percorrer a maior distância possível sem chocar contra obstáculos. O jogo decorre numa janela de texto. O pássaro estará sempre na zona esquerda da janela, deslocando-se apenas na vertical. No lado direito do ecrã vão surgindo obstáculos que se deslocam a uma velocidade constante na direção do pássaro, e que devem ser evitados.

Neste documento são descritos os detalhes de funcionamento pretendidos para o jogo. O jogo será programado usando a linguagem *assembly* do P3. O desenvolvimento e teste do programa serão realizados usando o simulador do P3 (p3sim), sendo usados os diversos recursos disponibilizados, de que se destacam: a janela de texto, os *displays* de 7 segmentos, os LEDs, o *display* LCD e os interruptores. O projeto deverá depois ser demonstrado no laboratório na placa com a versão hardware do P3.

2. Descrição do Jogo

2.1. Espaço de Jogo

O jogo irá desenrolar-se na Janela de Texto do simulador do P3. Esta janela corresponde a uma matriz de 80 colunas por 24 linhas em que, em cada posição, pode ser escrito um caráter ASCII.

Na janela de texto estão desenhadas duas linhas horizontais (a primeira e a última linha do ecrã) que delimitam o terreno de jogo. Estas linhas estão desenhadas recorrendo a uma sequência de carateres '-'.

O pássaro é formado por dois carateres "O>". Os obstáculos são representados pelo caráter "X". Os obstáculos devem aparecer na coluna mais à direita do ecrã, e deslocar-se na direção do pássaro, numa velocidade controlada pelo jogador, criando assim a ilusão do avanço do pássaro ao longo do percurso. O espaçamento entre obstáculos deve ser sempre de 5 carateres.

A Figura 1 apresenta o espaço de jogo no início do jogo.

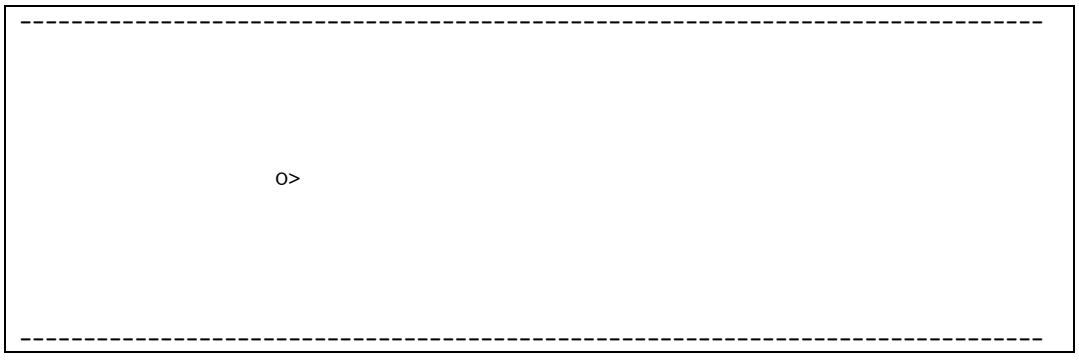

Figura 1 – Disposição inicial do espaço de jogo na Janela de Texto do P3

2.2. Pássaro

O pássaro deve começar na coluna 20 e na linha 12 (ver Figura 1). O pássaro deve-se deslocar apenas para cima e para baixo. Para isso, o jogador utilizará o botão de interrupção I0 para fazer mover o pássaro para cima a uma velocidade constante até subir um número constante de linhas, parâmetros estes (velocidade e número de linhas que sobe de cada vez que carrega no botão) que deverá escolher de forma a otimizar a experiência de jogo. O movimento para baixo (ou a desaceleração do movimento para cima) far-se-á sempre que o botão não tiver sido recentemente premido, de acordo com as leis da física (supondo a ausência de atrito). Note que estas velocidades estão dependentes da escala escolhida para o terreno de jogo, ou seja, a quantos metros corresponde uma linha. Deverá também escolher este parâmetro por forma a otimizar a experiência de jogo.

2.3. Obstáculos

Os obstáculos, simbolizados por um 'X', surgem na coluna da direita da Janela de Texto. Numa primeira fase, para uma cotação parcial, comece por colocar obstáculos de um caráter na parte inferior da janela (na linha 20), que o pássaro terá de saltar para os poder ultrapassar. Só após ter esta funcionalidade completa deverá, numa segunda fase e para a cotação completa, substituir estes obstáculos simples só com um caráter por duas colunas verticais de altura variável nas partes inferior e superior da janela, com um espaço de dimensão fixa a meio (o espaço estará a uma altura escolhida de forma aleatória), sendo que o pássaro deverá passar pelo meio delas. No caso de colisão com o pássaro, o jogo deve terminar e ser impressa no centro de ecrã uma mensagem de fim de jogo.
Os obstáculos, quando chegam à parte esquerda da janela de texto, desaparecem.

2.4. Dispositivos

Para além da funcionalidade principal do jogo na janela de texto, são usados também os seguintes dispositivos:

- o LCD deverá mostrar, na primeira linha, a distância atual percorrida:
  Distancia: XXXXX colunas
- os displays de 7 segmentos deverão indicar o número de obstáculos ultrapassados;
- os leds são usados para indicar o nível do jogo: os leds vão sendo progressivamente ligados da esquerda para a direita à medida que a velocidade do pássaro (e portanto o nível de dificuldade do jogo) aumenta. Por exemplo, no nível 1 está apenas ligado o led da esquerda; no nível 2 estão os dois leds mais à esquerda, e assim sucessivamente. Os botões I1 e I2 permitem diminuir e aumentar o nível do jogo, respetivamente. As velocidades (em número de colunas por segundo) associadas a cada nível ficam ao critério de cada grupo, desde que sejam crescentes.

2.5. Início do Jogo

Quando arranca, o programa deverá escrever, centrado respetivamente nas linhas 12 e 14, as mensagens "Prepare-se" e "Prima o interruptor I1".
Quando I1 for premido:

- a janela de texto deverá ficar como indicado na Figura 1;

- a pontuação atual deve ser colocada a 0;
- apenas o led mais à esquerda fica aceso (nível mais fácil);
- os displays de 7 segmentos devem mostrar o valor 0.
- é gerado um primeiro obstáculo, que depois terá o comportamento indicado na Secção 2.3;

2.6. Fim do Jogo

O jogo termina quando o pássaro chocar com um obstáculo. Nesta altura, o programa deverá escrever, centrado respetivamente nas linhas 12 e 14, as mensagens "Fim do Jogo" e a pontuação final (em obstáculos superados). Quando qualquer botão ou interruptor for premido o programa deve terminar.

3. Implementação

Uma vez iniciado, o jogo consistirá num ciclo em que se atualizam as posições do pássaro e dos obstáculos. Os movimentos são conseguidos apagando o símbolo da posição atual (i.e., escrevendo um espaço por cima) e escrevendo-o na nova posição.

3.1. Movimentação do pássaro

A posição do pássaro é atualizada constantemente, ou pelo premir do botão, ou pela ação da gravidade. Este movimento implica a atualização dos dois carateres que definem o pássaro. Naturalmente, é necessário verificar os limites da pista de jogo. Numa primeira fase, para uma cotação parcial, comece por considerar que a gravidade provoca uma deslocação para baixo com uma velocidade constante. Só após ter esta funcionalidade estável deverá então, numa segunda fase e para a cotação completa, considerar um movimento uniformemente acelerado. Note que o deslocamento por efeito da gravidade deve ser controlado pelo temporizador, mas que tal não corresponde a um deslocamento de uma linha por cada interrupção do temporizador, mas antes a uma fração de linha. (Sugestão: pode utilizar para a posição do pássaro um número de virgula fixa.)

3.2. Movimentação do obstáculo

A nova posição do obstáculo é sempre a posição à esquerda da atual. O ritmo de atualização da sua posição é determinado pela sua velocidade, com as temporizações indicadas na secção anterior. Quando o obstáculo atinge a coluna da esquerda da Janela de Texto, o obstáculo desaparece.

3.3. Temporizações

O ritmo de atualização das posições dos obstáculos é controlado através do temporizador disponível no simulador do P3. O temporizador terá por isso que ser programado por forma a permitir o ajuste do nível e ação da gravidade.
De notar que, apesar de o enunciado especificar ajustes na velocidade do pássaro, na verdade o que será temporizado será o movimento dos obstáculos. Outro aspeto importante é que a velocidade de queda do pássaro não pode depender do nível de dificuldade, e como tal não se pode usar diretamente o mesmo temporizador para atualizar tanto a posição dos obstáculos como a posição do pássaro. Em vez disso, sugere-se configurar o temporizador para um intervalo de tempo fixo, e utilizar um contador para controlar a movimentação dos obstáculos (o contador vai de 0 a N-1 e provoca a movimentação dos obstáculos sempre que der a volta), ao passo que o controlo da distância de queda do pássaro pode ser feito calculando, para cada intervalo do temporizador, a fração de linha que o pássaro percorre.
Note que em geral as interrupções devem ser rotinas muito curtas, normalmente limitando-se a ajustar o valor de uma ou mais variáveis, que depois serão utilizadas para controlar a execução do programa principal.

3.4. Valores aleatórios

Quando aparece um novo obstáculo é necessário obter um valor aleatório para saber em que linha o espaço entre as colunas irá ficar. O Anexo A descreve um algoritmo para a geração de valores aleatórios de 16 bits.

3.5. Dúvidas na especificação do enunciado

Sempre que o enunciado não defina completamente como deve implementar alguma parte da solução, deve tomar a decisão que achar mais apropriada, tendo sempre em mente o objetivo de melhorar a experiência de jogo. Estas decisões devem ser descritas e justificadas no relatório final.

4. Plano de Desenvolvimento

4.1. Desenvolvimento do trabalho

Sugere-se o seguinte plano de desenvolvimento:

1. O primeiro passo é entender a estrutura geral da aplicação. Em particular, esta deverá ter, após o início do jogo, um ciclo de jogo cujas mudanças no estado do jogo ocorrem pela mudança de valor de uma dada variável, que por sua vez é modificada pela rotina de tratamento da interrupção do temporizador. Esta estrutura geral deve ser discutida entre os elementos do grupo, e se necessário também com o docente dos laboratórios, para ser bem entendida.
2. Desenhe o fluxograma que descreve cada um dos procedimentos da aplicação, com especial atenção à relação entre o fluxo do programa principal e as várias rotinas de tratamento de interrupção. Estes fluxogramas e a lógica funcional do programa principal deverão ser apresentados na primeira aula de projeto (conforme a calendarização no final do enunciado).
3. Para o conjunto de procedimentos que definiu, identifique claramente os parâmetros de entrada, as saídas (valores de retorno) e os registos modificados na sua execução.
4. Programe e teste minuciosamente cada uma das rotinas que efetuam a interface com os dispositivos de entrada (interruptores) e os dispositivos de saída (janela de texto, LEDs, display 7 segmentos e LCD), com especial atenção à passagem de parâmetros entre estas rotinas e o programa principal.
5. Configure o temporizador disponibilizado pelo simulador e associe o vector de interrupção respetivo com a rotina a executar periodicamente. Configure o ciclo de jogo com a estrutura atrás indicada, em que a mudança de valor de uma variável desencadeia uma atualização no estado do jogo.
6. Realize a ligação entre os vários procedimentos, de forma a obter o comportamento desejado e especificado.
7. Comente e indente devidamente o código desenvolvido. Inclua nos comentários referências aos fluxogramas que irá entregar para auxiliar a leitura e compreensão do programa.

4.2. Faseamento da codificação

Não deve tentar codificar todo o programa de uma só vez. Implemente as várias funcionalidades do programa de uma forma faseada e efetue os testes necessários para verificar o seu correto funcionamento. Não prossiga para a implementação de funcionalidades mais avançadas sem ter garantido que as que lhe servem de base estão corretamente implementadas.

Estando o sistema a funcionar corretamente, pode incluir eventuais funções opcionais que entretanto tenha desenvolvido.

5. Plano de Entrega

Semana de 9 de Novembro, na aula de laboratório	
Entrega: (em papel)	• Fluxogramas da aplicação e dos principais procedimentos que lhe servem de base (mesmo que ainda não estejam implementados), com especial atenção à relação entre o fluxo do programa principal, os vários módulos funcionais e as rotinas de tratamento de interrupção.
Demonstração:	• O programa a apresentar deverá fazer o desenho do cenário de jogo, bem como a colocação do pássaro e o seu movimento para cima (não é necessário implementar a gravidade).
Dia 2 de Dezembro, até às 15h00, na Sala de Estudo do DEI	
Entrega: (em papel)	• Código desenvolvido devidamente comentado e indentado (impresso frente e verso a duas páginas por face).
Dia 2 de Dezembro, até às 23h59, no Fénix	
Entrega: (electrónica)	A entrega final deverá ser submetida num ficheiro zip com o nome no formato tAaBBgC.zip, em que: A é o dia da semana (2 a 6); BB é a hora de início (basta a hora, com 2 digitos); C é o número do grupo. Este ficheiro deverá conter: • Breve relatório em PDF (máximo 2 páginas em 11pt) com a descrição do projeto realizado, organização do programa e explicação dos aspectos mais relevantes da implementação. Na conclusão deverá ser feito um balanço do que foi realizado, com indicação dos aspectos nos quais o projeto tenha divergido do enunciado base (funcionalidades adicionais implementadas, funcionalidades não implementadas, outras variações ou divergências, etc.). • Fluxogramas finais da aplicação e dos principais procedimentos que lhe servem de base. • Código, quer ficheiro fonte, quer um PDF gerado com a aplicação p3print fornecida na página da cadeira.
Semana de 7 de Dezembro, na aula de laboratório	
Demonstração:	• Funcionamento do jogo concebido.

As duas entregas em papel devem ser feitas num envelope identificado com o dia e hora do turno e número do grupo.
O calendário das discussões será acordado com o docente do turno respectivo. A discussão terá lugar, preferencialmente, na semana de 14 de Dezembro.

Anexo A – Geração de sequência pseudoaleatória

O seguinte algoritmo gera uma sequência aparentemente aleatória de números de 16 bits, com distribuição uniforme (isto é, os números são equiprováveis), com um passo de repetição elevado:

```
Máscara = 1000 0000 0001 0110 b
if(n0 =0) /*Testa o bit de menor peso*/
    Ni+1 = rotate_right (Ni);
else
    Ni+1 = rotate_right (XOR (Ni, Mascara));
```

Em cada invocação desta função lê-se o valor anterior N_i e gera-se um novo valor pseudoaleatório, N_{i+1}. A raiz desta sequência ($N_0 \neq 0$) pode ser obtida a partir de um parâmetro que varie com o decorrer do jogo (por exemplo, o número de ciclos de programa ou ciclos de espera entre a inicialização do programa e o início efetivo do jogo). Não deve modificar o valor de N_i entre invocações. Para obter um valor aleatório entre 0 e M pode dividir o valor de N_i por M e retirar o resto, mas sempre sem modificar o valor de N_i.