Aventura da Caça ao Tesouro

Preâmbulo

O trabalho é individual e é entregue no Mooshak, que aceita submissões até às 20:00 do dia 26 de outubro de 2024. Leia este enunciado com a máxima atenção, para perceber muito bem o problema e todos os detalhes sobre o concurso do Mooshak e os critérios de avaliação do trabalho.

1. Conceitos e Objetivos do Trabalho

Neste projeto, o objetivo é desenvolver um jogo interativo de caça ao tesouro numa masmorra de dois níveis. O jogador desloca-se para a esquerda ou para a direita para recolher tesouros, evitando os inimigos. Para ganhar o jogo, o jogador deve recolher todos os tesouros, navegar pela masmorra e chegar à saída. O jogo incluirá dois tipos de inimigos com padrões de movimento distintos, um sistema de escadas para se deslocar entre níveis e regras de movimento específicas que restringem a capacidade do jogador de se deslocar para fora dos limites da masmorra.

Configuração do jogo

A masmorra é composta por dois níveis, cada um com uma sequência de salas, como mostra a Figura 1. Cada sala é identificada pela sua posição na sequência, atribuída da esquerda para a direita (ou seja, a primeira célula ou a célula na primeira posição, a segunda célula ou a célula na segunda posição, etc.). A dimensão de cada nível pode variar e é determinada pelo número de salas. Os dois níveis estão ligados por uma escada que permite ao jogador deslocar-se entre níveis sendo colocado na sala correspondente da escada.

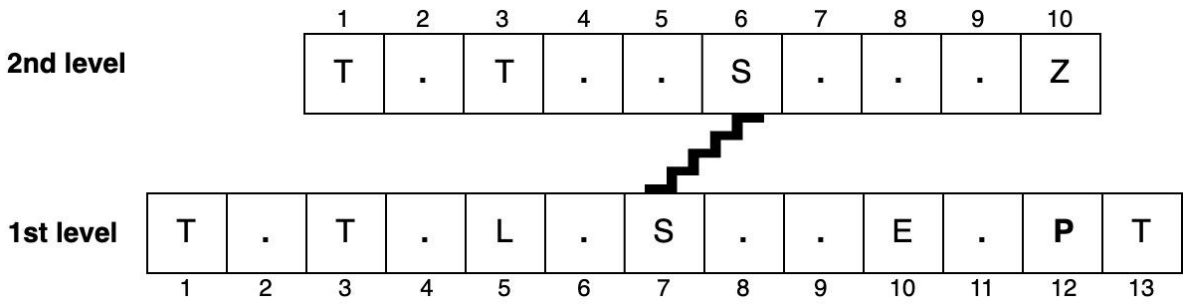

Figura 1: Exemplo de uma configuração da masmorra.

Os caracteres seguintes são utilizados para representar os diferentes elementos da masmorra:

- ‘P’ representa a sala de entrada. O jogador é colocado nesta sala no início do jogo.

- ‘E’ representa a sala de saída. Para ganhar o jogador deve chegar a esta sala depois de recolher todos os tesouros.
- '.' (ponto final) representa uma sala vazia. O jogador pode passar em segurança por estas salas.
- ‘T’ representa uma sala com um tesouro. O jogador deve recolher todos os tesouros para ganhar o jogo.
- ‘S’ representa a escada. Se o jogador se mover para uma escada, passa automaticamente para a escada correspondente no outro nível.
- ‘L’ representa a posição de um inimigo ***looper*** que se desloca numa trajetória circular para a direita.
- ‘Z’ representa a posição de um inimigo ***zigzagger***, que se desloca em ziguezague, alternando o movimento para a direita com passos crescentes.

A masmorra tem de ter dois níveis e incluir uma única sala de entrada e uma única sala de saída, um inimigo em cada nível, uma escada que liga os níveis e pelo menos um tesouro.

Na Figura 1, no primeiro nível, há tesouros nas salas 1, 3 e 13; há um inimigo *looper* na sala 5, uma escada na sala 7, uma saída na sala 10 e o jogador começa o jogo na sala 12. No segundo nível, a escada está na sala 6, há tesouros nas salas 1 e 3, e há um inimigo *zigzagger* na sala 10. As restantes salas de ambos os níveis estão vazias.

Movimento do Jogador e Limites

O jogador move-se para a esquerda ou para a direita especificando a direção e o número de passos que pretende dar. Por exemplo, se o jogador indicar o comando “right 3”, vai tentar mover (saltar) três passos para a direita. Se o movimento do jogador o colocar para além dos limites do nível, o sistema move-o automaticamente para o mais longe possível dentro dos limites do nível, ou seja, o movimento é um salto. Por exemplo, se o jogador estiver na sala mais à esquerda do nível (sala 1) e tentar mover-se para a esquerda, permanecerá na mesma sala. Da mesma forma, permanecerá na última sala se tentar mover-se para a direita para além dessa sala.

Inimigos e o seu Comportamento

O jogo pode incluir dois tipos de inimigos, cada um com padrões de movimento distintos. Cada nível inclui um único inimigo que pode ser de qualquer um dos tipos. Ambos os inimigos seguem um movimento circular, ou seja, a última sala de cada nível pode ser vista como ligada à primeira sala desse mesmo nível. Por exemplo, na Figura 1, se um inimigo estiver na sala 13 e se deslocar 3 passos para a direita, deslocar-se-á para a sala 3.

- Inimigo *looper*: este inimigo desloca-se um passo para a direita em cada jogada. Quando chega à última sala de um nível, como explicado acima, volta à primeira sala e continua o seu movimento circular.
- Inimigo *zigzagger*: este inimigo segue um padrão de movimento mais imprevisível. Move-se um passo para a direita, dois passos para a direita, três passos para a direita, quatro para a direita e, por fim, cinco para a direita. Depois, reinicia o número de passos,

e começa com um passo para a direita, e assim por diante. Se, em qualquer passo, o ziguezague atingir os limites do nível, dá a volta num movimento circular.

No jogo, os inimigos e os jogadores movem-se simultaneamente de acordo com os seus respectivos padrões. Depois de todas as personagens se terem movimentado e do jogador, o jogo verifica se o jogador e um inimigo se deslocaram para a mesma sala. Se estiverem na mesma sala, o jogo termina e o jogador perde. No caso de derrota é irrelevante se a sala tem ou não um tesouro e também é irrelevante se é a sala de saída. Para simular o movimento simultâneo das várias personagens, o jogador move-se primeiro, seguido do inimigo no nível 1 e depois do inimigo no nível 2. Só então o jogo deve verificar se o jogador e um inimigo estão na mesma sala.

Coleção de tesouros

O objetivo principal do jogador é recolher todos os tesouros dos dois níveis. Quando o jogador salta para uma sala com um tesouro, recolhe-o automaticamente. Uma vez recolhido, o tesouro é retirado do jogo. O jogador não ganha o jogo até ter recolhido todos os tesouros de ambos os níveis. É importante notar que os inimigos não têm qualquer efeito sobre os tesouros, o que significa que os inimigos não recolhem tesouros.

Sistema de Escadas

Cada nível tem uma escada que o liga ao outro nível. Quando o jogador se movimenta para uma a escada, desloca-se automaticamente para a escada correspondente no outro nível.

- Se o jogador estiver no nível 1 e se movimentar para a sala com a escada, é transportado para a escada do nível 2.
- Se o jogador estiver no nível 2 e se movimentar para a sala com a escada, é transportado para a escada do nível 1.
- As escadas são pontos de transição entre níveis para o jogador e não têm efeito nos inimigos, pois estes não podem transitar entre níveis.

A súbita e descida de escadas é automático, o que significa que, assim que o jogador salta para uma sala com uma escada, é imediatamente transportado para a escada do outro nível.

Condições de término do jogo

O jogo termina quando o jogador recolhe todos os tesouros e chega à sala de saída. Nesta altura, o jogador ganha o jogo. No entanto, se o jogador e o inimigo se moverem para a mesma sala, o jogador é apanhado, e o jogo termina com uma derrota.

Quando o jogo termina, quer o jogador tenha ganho ou perdido, o jogador e os dois inimigos não voltam a mover-se.

Como Jogar

No jogo, o jogador pode introduzir comandos para se deslocar para a esquerda ou para a direita num determinado número de passos. Por exemplo, “left 2” é usado para mover dois passos para a esquerda e “right 3” é usado para mover três passos para a direita. Os inimigos movem-se depois de o jogador se ter movido de acordo com os seus padrões predefinidos.

2. Especificação do Sistema

Pretende-se que a interface da aplicação seja simples, para poder ser utilizada em ambientes diversos e permitir automatizar o processo de teste. Por estes motivos, a entrada e a saída têm de respeitar o formato preciso especificado nesta secção. **Pode assumir que o input obedece às restrições de valor e de formato indicadas**, ou seja, que o utilizador não comete erros, para além dos previstos neste enunciado.

O programa lê linhas da entrada padrão (`System.in`), e escreve linhas na saída padrão (`System.out`) e distingue maiúsculas de minúsculas (por exemplo, as palavras "right" e "Right" são diferentes).

Formato do Input

A entrada tem a seguinte estrutura (onde o símbolo ↵ representa uma mudança de linha):

layoutDungeonLevel$_2$↵
layoutDungeonLevel$_1$↵
direction steps↵
direction steps↵
...
direction steps↵
quit

onde :

- **layoutDungeonLevel$_1$** e **layoutDungeonLevel$_2$** são duas sequências de caracteres com comprimentos possivelmente diferentes (que são números entre 4 e 100). Cada elemento da sequência é um dos caracteres maiúsculos 'P', 'E', 'S', 'T', 'L', 'Z' ou o carácter '.' (ponto).
- **direction** é "left" (esquerda) ou "right" (direita), e **steps** é um número inteiro positivo. Define o *comando-movimento* do jogo.

A primeira e a segunda linhas da entrada especificam, respectivamente, a disposição do segundo e do primeiro nível da masmorra. Segue-se um número arbitrário de movimentos. A última linha contém um comando especial, o *comando-sair*, que só pode ocorrer na última linha porque termina a execução do programa.

Comando-movimento

O *comando-movimento* indica que se pretende deslocar o jogador para uma nova posição. Já vimos que as linhas com *comando-movimento* têm o seguinte formato (com um espaço a separar as duas componentes):

direction steps↵

onde a **direction** é 'left' ou 'right' e **steps** é um número inteiro positivo.
O programa escreve uma linha na consola, distinguindo quatro casos:

- Se o jogo já tiver terminado, a linha tem:

  `The game is over`↵

- Se o jogador se deslocar para a mesma sala de um inimigo:

  `You lost the game!`↵

- Se o jogador se deslocar para a sala de saída e tiver recolhido todos os tesouros:

  `You won the game!`↵

- Nos restantes casos, a linha tem o seguinte formato:

  `Player: level num`*`Level`*`, room `*`numRoom`*`, treasures `*`numTreasures`↵
  `Level 1 enemy: `*`enemyType`*`, room `*`numRoom`↵
  `Level 2 enemy: `*`enemyType`*`, room `*`numRoom`↵

  em que **`num`*`Level`*** indica o nível em que jogador se encontra, ***`numRoom`*** indica a sala em que a personagem do jogo se encontra, ***`numTreasures`*** indica quantos tesouros já foram recolhidos pelo jogador e ***`enemyType`*** descreve o tipo de inimigo, que pode ser *looper* ou *zigzagger*.

Command-sair

O comando *commando-sair* indica que se pretende terminar a execução do programa. A linha com o *commando-sair* tem:

`quit`↵

O programa termina, escrevendo uma linha na consola. Distinguem-se três casos:

- Se o jogo ainda não tiver terminado, a linha escrita é:

  `The game was not over yet!`↵

- Se o jogo tiver terminado e o jogador tiver ganho o jogo, a linha escrita é

  `Goodbye: You won the game!`↵

- Se o jogo tiver terminado e o jogador tiver perdido o jogo, a linha escrita é

  `Goodbye: You lost the game!`↵

Comandos Invalidos

Sempre que o utilizador escrever uma linha que não comece com as palavras “right”, “left” ou “quit” o estado do jogo não deve ser alterado e o programa deve escrever uma linha com:

`Invalid command`↵

3. Exemplos

Apresentam-se alguns exemplos. A coluna da esquerda ilustra a interação: o input está escrito a **`blue`** e o output a **`black`**. Todas as linhas do input e do output terminam com o símbolo de mudança de linha, que se omitiu para aumentar a legibilidade. A coluna da direita tem informação para o leitor do enunciado, servindo apenas para relembrar as regras descritas anteriormente.

Exemplo 1

```
S.T..L.E
P.L...T.S
right 2
Player: level 1, room 3, treasures 0
Level 1 enemy: looper, room 4
Level 2 enemy: looper, room 7
right 5
Player: level 1, room 8, treasures 0
Level 1 enemy: looper, room 5
Level 2 enemy: looper, room 8
left 1
Player: level 1, room 7, treasures 1
Level 1 enemy: looper, room 6
Level 2 enemy: looper, room 1
right 2
Player: level 2, room 1, treasures 1
Level 1 enemy: looper, room 7
Level 2 enemy: looper, room 2
right 2
You lost the game!
right 2
The game is over
quit
Goodbye: You lost the game!
```

Masmorra do nível 2 com 8 salas.
Masmorra do nível 1 com 9 salas.
Os *loopers* movem-se 1 posição para a direita.
O inimigo no nível 2 move-se 1 posição para a direita e passa para a sala 1.
O jogador desloca-se para a escada da sala 9 e é transportado para a escada do nível 2 (sala 1).
O jogador vai para a sala 3 no nível 2, mas é aqui que o inimigo looper está agora localizado.

Example 2

```
..TT..S.Z..
L.TPSE.T
jump 4
Invalid command
left 1
Player: level 1, room 3, treasures 1
Level 1 enemy: looper, room 2
Level 2 enemy: zigzagger, room 10
right 2
Player: level 2, room 7, treasures 1
Level 1 enemy: looper, room 3
Level 2 enemy: zigzagger, room 1
left 4
Player: level 2, room 3, treasures 2
Level 1 enemy: looper, room 4
Level 2 enemy: zigzagger, room 4
right 1
```

Masmorra do nível 2 com 11 salas.
Masmorra do nível 1 com 8 salas.
Looper move-se 1 posição para a direita.
Zigzagger move-se 1 posição para a direita.
O jogador sobe as escadas para a sala 7 no nível 2
Zigzagger desloca-se 2 posições para a direita e salta para a sala 1.
Zigzagger move-se 3 posições para a direita.

```
Player: level 2, room 4, treasures 3
Level 1 enemy: looper, room 5
Level 2 enemy: zigzagger, room 8
```
Zigzagger move-se 4 posições para a direita.

```
right 3
Player: level 1, room 5, treasures 3
Level 1 enemy: looper, room 6
Level 2 enemy: zigzagger, room 2
```
O jogador desce as escadas para a sala 4 no nível 1. *Zigzagger* move-se 5 posições para a direita e salta para a sala 2.

```
right 3
Player: level 1, room 8, treasures 4
Level 1 enemy: looper, room 7
Level 2 enemy: zigzagger, room 3
```
Zigzagger move-se 1 posição para a direita.

```
left 2
You won the game!
```
O jogador já recolheu todos os tesouros e move-se para a saída.

```
quit
Goodbye: You won the game!
```

4. Entrega do Trabalho

O trabalho é entregue no Mooshak. Deverá submeter um arquivo .zip ao Problema A do concurso **`IP2425-P1`**. Não se esqueça que:

- O arquivo deve conter apenas todos os ficheiros `.java` que tiver criado para resolver o problema.
- O arquivo tem necessariamente de conter um ficheiro `Main.java`, onde está o método `main`.
- A classe `Main` tem de pertencer ao pacote principal (default package).
- A versão de Java instalada no Mooshak é a 21

Cada aluno receberá as credenciais de acesso ao concurso IP2425-P1 (que deverão ser diferentes das credenciais de acesso ao concurso IP2425-Aulas), no seu endereço de email institucional.
O concurso IP2425-P1 abre no dia 16 de outubro e encerra às 20h00 do dia 26 de outubro de 2024 (sábado). Pode ressubmeter o trabalho as vezes que entender, até à hora limite de submissão. Apenas será avaliado o programa que obtiver a maior pontuação no Mooshak; se houver vários programas com a maior pontuação, será avaliado, de entre esses, o último que tiver sido submetido. Se quiser que o programa avaliado seja outro, tem de enviar uma mensagem à Prof. Carla Ferreira (carla.ferreira@fct.unl.pt) até uma hora após o concurso fechar, indicando o número da submissão que pretende que seja avaliada.

5. Critérios de Avaliação do Trabalho

De acordo com o Regulamento de Avaliação de Conhecimentos da FCT NOVA:

1. Existe fraude quando:
    a. Se utiliza ou tenta utilizar, sob qualquer forma, num teste, exame, ou outra forma de avaliação, ferramentas, informações e/ou outros equipamentos não autorizados pelo Regente da UC;

   b. Se presta ou recebe colaboração não autorizada na realização dos exames, testes, ou qualquer outra prova de avaliação de conhecimentos individuais;
   c. Se presta ou recebe colaboração, não permitida pelas regras aplicáveis, a cada caso na realização de trabalhos práticos, relatórios ou outros elementos de avaliação.
2. Existe plágio quando se omite a consulta de fontes ou a ajuda de outrem nos documentos produzidos, nomeadamente no que se refere à elaboração de trabalhos.
3. Quando for comprovada a existência de fraude ou plágio, em qualquer dos elementos de avaliação de uma UC, os estudantes diretamente envolvidos são liminarmente reprovados na UC, sem prejuízo de eventual procedimento disciplinar ou cível, devendo a ocorrência ser participada ao Diretor da NOVA FCT pelo Responsável da UC.
4. Em caso de dúvida, o Responsável, ou o Regente, da UC pode determinar a realização de uma nova prova, escrita ou oral, equivalente, cujo resultado prevalece relativamente a outro anteriormente obtido.

Em IP, o documento Colaboração Permitida e Não Permitida, disponibilizado no Moodle, clarifica as alíneas transcritas acima. Os alunos que cometerem fraude num trabalho não obterão frequência.

A avaliação do trabalho tem duas componentes independentes, cujas notas se somam para obter a nota do trabalho:

- **Funcionalidade** (correção dos resultados produzidos): 1**0 valores**

  Um programa submetido ao concurso que só use as classes da biblioteca permitidas e que obtenha P pontos no Mooshak teré P/10 valores[1]. Se o programa usar classes da biblioteca não permitidas, a nota da funcionalidade será inferior a P/10 valores.

  As classes de biblioteca permitidas são as seguintes: `System.in`, `System.out`, `Scanner`, `String` e `Math`.

  Serão fornecidos testes de treino, semelhantes aos utilizados pelo Mooshak. Se o programa produzir os resultados corretos com todos os testes de treino, é provável (mas não é garantido) que obtenha 100 pontos no Mooshak.

- **Qualidade do código: 10 valores[2]**

  Um código com qualidade tem, entre outras, as seguintes características:

  - Separação, em diferentes métodos, entre os métodos de interação com o utilizador e os métodos do domínio do problema.
    - Por exemplo, os métodos que lêem ou escrevem (ou seja, interagem com o utilizador) não devem alterar o estado do jogo, revelar qualquer parte da lógica do jogo ou mostrar a forma como o jogo é implementado;
  - Siga a estrutura do programa apresentada nas aulas: em primeiro lugar, defina as constantes do programa, em segundo lugar as variáveis globais, em terceiro lugar os métodos de domínio, em quarto lugar os métodos de interação e, por último, o método principal;

[1] O trabalho pode ser realizado incrementalmente. Por exemplo, pode começar por assumir que só existem inimigos looper. E claro que, enquanto o programa não produzir os resultados corretos em todos os testes do Mooshak, não obterá 100 pontos.

[2] Note que a qualidade do código tem um peso muito grande na nota do trabalho.

- Métodos, variáveis e constantes com objetivos bem definidos;
- Algoritmos simples e bem estruturados, implementados com as instruções mais adequadas;
- Identificadores que expressam os conceitos que representam, escritos de acordo com as convenções ensinadas (por exemplo, o nome de um método deve ser um verbo que começa com uma letra minúscula);
- Indentação correta com linhas com 100 caracteres (no máximo) e métodos com 25 linhas (no máximo);
- Um comentário antes de cada método (à exceção do método main) que indique resumidamente o que o método faz.

Boa sorte!