

Escola Superior de Tecnologia e Gestão de Beja  
Curso de Engenharia Informática

## **Sistemas Operativos**

### **Trabalho de Grupo N.º 1**

*Programação de um Jogo de Aventuras*

Luís Garcia

## ***Programação de um Jogo de Aventuras***

Neste trabalho pretende-se que os alunos desenvolvam um jogo de aventuras utilizando os conhecimentos de programação sistema adquiridos nas aulas.

Os jogos de aventuras tiveram provavelmente a sua origem no jogo de tabuleiro *Dungeons & Dragons (D&D)* (<http://www.ddo.com/>) onde os jogadores encarnam personagens que percorrem territórios em busca de tesouros e enfrentam diversas criaturas que se cruzam no seu caminho.

O primeiro jogo de computador deste tipo foi o “*Colossal Cave Adventure*” (<http://www.rickadams.org/adventure/>) desenvolvido por Willie Crowther e Don Woods (<http://www.icynic.com/~don/>). Um outro destes primeiros jogos foi o Zork (<http://www.infocom-if.org/downloads/downloads.html>)

O jogo que se pretende que os alunos desenvolvam neste trabalho deverá conter quatro elementos fundamentais: (1) o Jogador; (2) o Local da Aventura; (3) o Monstro e (4) o Tesouro. Em seguida iremos sugerir uma forma para a criação destes elementos num programa de computador.

O Jogador poderá consistir numa variável do tipo estrutura. Deverá conter pelo menos os seguintes campos: (1) nome, (2) energia, (3) local onde se encontra e (4) a indicação sobre se já possui ou não o tesouro (0 – não possui; 1 – possui).

O Local da Aventura poderá ser representado por um vector de elementos do tipo Sala ou Célula. Esta última denominação é mais adequada se existirem espaços exteriores. Cada elemento deste

vector será uma Sala ou Célula com os seguintes campos: (1) norte; (2) sul; (3) oeste; (4) este; (5) descrição e (4) a existência ou não do tesouro. Os campos norte, sul, oeste e este irão conter o número da Sala ou Célula para onde o Jogador se deslocará se decidir ir naquela direcção. Quando não existir uma passagem numa determinada direcção o campo poderá conter o valor -1. A descrição será uma cadeia de caracteres (*string*) que fornecerá uma breve descrição sobre o local onde o jogador se encontra. O último campo indicará se existe ou não um tesouro naquele local (0 – não existe; 1 – existe).

O Monstro poderá ser representado através de uma estrutura com dois campos: (1) energia e (2) local onde se encontra.

Normalmente o objectivo destes jogos consiste na obtenção do tesouro e regresso ao ponto de partida ou identificação de uma saída onde terminará a aventura. No entanto durante a exploração do local da aventura o Jogador poderá ter de enfrentar alguns monstros...

Dependendo do tema do trabalho escolhido pelos alunos o local da aventura, o tesouro e o monstro (ou monstros) poderão assumir diferentes formas. A criação de um ambiente original também será um aspecto valorizado no trabalho.

O pseudo-código básico de um programa de aventuras deste género pode ser expresso na forma apresentada a seguir.

```

Inicialização do Jogador
Inicialização do Local da Aventura
Inicialização do Monstro
Enquanto não for Fim de Jogo
{
    Movimentar Monstro
    Descrever a Localização do Jogador
    Aceitar Comando do Jogador
    Movimentar Jogador
    Se a Localização do Jogador for igual à do Monstro
        Lutar
}
Apresentar Resultado Final

```

**Importante:** Os alunos devem desenvolver o programa de forma gradual e testar cada função desenvolvida. Por exemplo, ao criarem uma função para a inicialização do Jogador devem também criar uma função para listar o Jogador, que será chamada na fase de testes para verificação do funcionamento correcto da função de inicialização do Jogador. É importante que os alunos sigam uma abordagem deste género, pois sem a realização de testes metódicos irão deixando diversas falhas no código produzido (e em pontos distintos) que dificultarão o desenvolvimento do programa. É importante que em cada momento todas as funções desenvolvidas anteriormente estejam correctas e devidamente testadas. Apenas a função em desenvolvimento poderá apresentar problemas. Assim o aluno saberá onde se encontra a falha que deverá corrigir.

Este programa deverá ser desenvolvido com a linguagem C/C++ e o ambiente de desenvolvimento Visual Studio (ou outro para o ambiente Windows).

Para este trabalho propõe-se a realização das seguintes tarefas ao longo das próximas semanas. As tarefas a executar não estão descritas na totalidade pelo que é da responsabilidade dos alunos a resolução das situações omissas.

## **Semana 1 e Semana 2**

(5 + 1 valores)

1. Implementação de uma versão simples do jogo. Os alunos devem criar um ambiente original para a acção do jogo. Este aspecto também será valorizado na avaliação do trabalho (1 valor). Para a implementação desta primeira versão do jogo pode ser seguida a seguinte estrutura:

*Inicialização do Local da Aventura*

*Inicialização do Jogador*

*Inicialização do Monstro*

*Enquanto não for Fim de Jogo{*

*Movimentar Monstro*

*Descrever a Localização do Jogador*

*Aceitar Comando do Jogador*

*Movimentar Jogador*

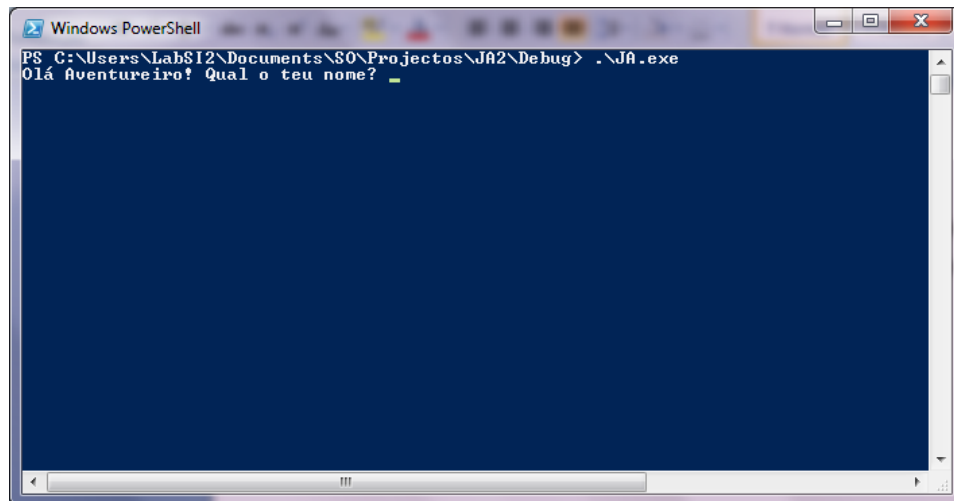
*Se a Localização do Jogador for igual à do Monstro*

*Lutar*

*}*

*Apresentar Resultado Final*

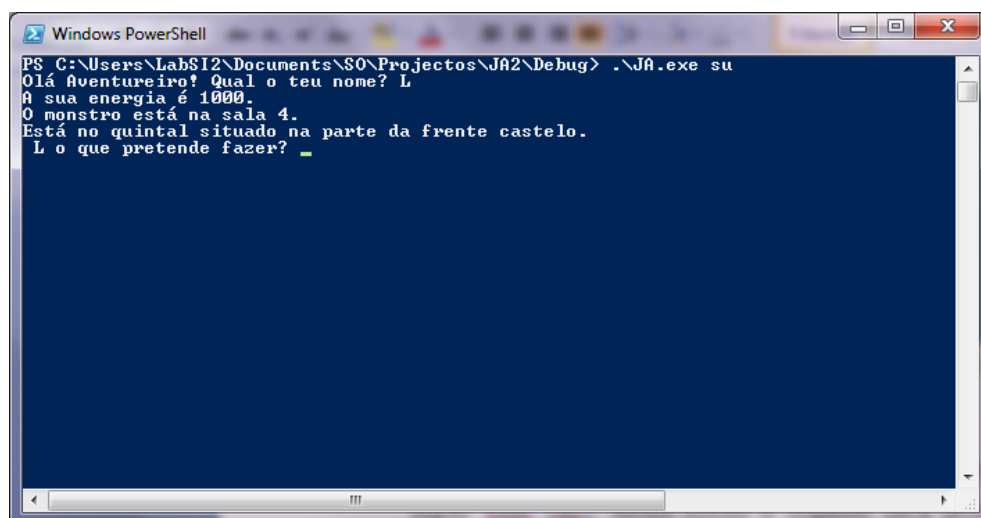
O Jogo de Aventura (JA) deverá ser chamado na linha de comando como se apresenta a seguir.



```
Windows PowerShell
PS C:\Users\LabSI2\Documents\S0\Projectos\JA2\Debug> .\JA.exe
Olá Aventureiro! Qual o teu nome? _
```

(1 valor)

2. Para testar devidamente o programa deve permitir que este seja chamado em modo *super user* (*su*). Neste modo o Jogador terá uma energia muito superior à energia do Monstro (100 vs 10 respectivamente). Também deverá apresentar em cada jogada a localização do Monstro. As probabilidades do Monstro ganhar uma luta também poderão ser diminuídas. Para jogar neste modo o programa será chamado da forma apresentada a seguir.

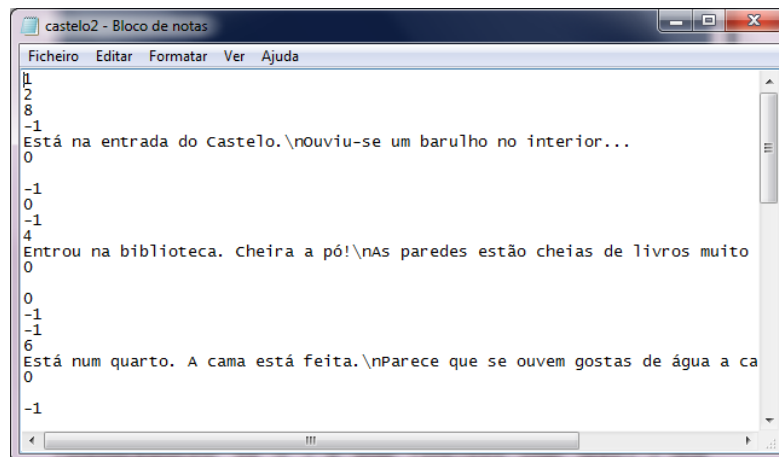


```
Windows PowerShell
PS C:\Users\LabSI2\Documents\S0\Projectos\JA2\Debug> .\JA.exe su
Olá Aventureiro! Qual o teu nome? L
A sua energia é 1000.
O monstro está na sala 4.
Está no quintal situado na parte da frente castelo.
L o que pretende fazer? _
```

## Semana 3

(2 valores)

3. Inicialize as Salas ou Células do Local da Aventura a partir de um ficheiro de texto. O ficheiro poderá ter a seguinte estrutura:



Neste exemplo o primeiro bloco de dados diz respeito à Sala 0, o seguinte à Sala 1 e assim por diante. Cada bloco contém as salas destino para cada uma das direcções (n, s, o, e), uma descrição da sala e a indicação se contém ou não o tesouro (0 – não contém ou 1 - contém).

(2 valores)

4. Inicialize as Salas ou Células do Local da Aventura a partir de um ficheiro binário. Desta forma tornará o conteúdo deste ficheiro inacessível através de um editor de texto.

## Semana 4

(2 valores)

5. Desenvolva um novo comando que permita gravar a situação de jogo que poderá ser retomada mais tarde.

(2 valores)

6. Melhore a interface do programa utilizando as funções da API do Windows para manipulação da consola.

(2 valores)

7. Podem também ser desenvolvidas outras funcionalidades extra não identificadas neste enunciado.

## **Semana 5**

(2 valores)

8. Identifique no programa funções com as quais possa ser constituída uma biblioteca dinâmica. Estas funções podem consistir por exemplo num conjunto de funções utilitárias que possam ser úteis noutros programas deste género. Também pode criar uma *Dll* com funções relacionadas, por exemplo escrita na consola, que preveja virem a sofrer actualizações em versões futuras. Assim, para a actualização do programa bastará a substituição da *Dll*. Para experimentar a *Dll* crie uma nova versão do programa. Aconselha-se a inclusão da MFC neste novo projecto de consola para que a ligação à *Dll* seja mais simples.

(1 valor)

9. Realize uma apresentação e relatório do trabalho.

## **Avaliação**

Os trabalhos serão avaliados de acordo com a quantidade e qualidade das funcionalidades implementadas. Apenas serão aceites trabalhos desenvolvidos parcialmente nas aulas da disciplina. Não são aceites soluções desenvolvidas exclusivamente fora das aulas. Os vários elementos do grupo devem participar na elaboração do trabalho. **Não**



**serão toleradas cópias nas soluções apresentadas. Nestas situações os alunos obterão nota zero no trabalho.**

## **Grupos de Trabalho**

O trabalho deve ser desenvolvido por grupos com um máximo de dois elementos. Não são permitidas alterações nos elementos do grupo salvo em situações extraordinárias e devidamente justificadas.

## **Entrega do Trabalho**

Os alunos devem entregar o ficheiro executável com o programa, um pequeno relatório que apresente a solução encontrada e a apresentação electrónica do trabalho. No relatório devem ser apresentadas e explicadas as principais partes do código. Num anexo deste relatório deve ser colocada a totalidade do código da aplicação. Para a entrega do trabalho os alunos devem compactar o ficheiro executável, relatório e apresentação num único ficheiro (zip ou equivalente). O nome deste ficheiro deve conter a indicação TG1 (Trabalho de Grupo 1) e o número do grupo de trabalho. Por exemplo TG1\_03.zip seria o Trabalho de Grupo 1 do grupo com o número 03. O trabalho deve ser entregue através do sistema *Moodle*. **Não serão consideradas soluções entregues por e-mail.**

## **Apresentação do Trabalho**

Os alunos deverão apresentar a solução desenvolvida ao docente. Esta apresentação poderá ser efectuada no decorrer de uma aula ou se necessário num horário especialmente marcado para o efeito.

*Bom Trabalho*

*Luís Garcia*