Programação I

Relatório do Trabalho Prático

Implementação do jogo “A Poção Mágica”

2013/2014

-João Aldeano 30395

-José Rocha 31149

-Gabriel Charrua 32457

Introdução

No âmbito da cadeira de Programação I, realizou-se um trabalho que consistiu no desenvolvimento e implementação do jogo "A Poção Mágica" em liguagem de programação Python.

Desenvolvimento/Implementação

Começámos por implentar as variáveis globais que são os inputs introduzidos pelo utilizador (Opção 1) ou a leitura do ficheiro .txt (Opção 2), dependendo do 1º input do utilizador.

Seguidamente criámos as funções que:

-Cria o tabuleiro N x N preenchido por 0's (cria_tabuleiro)

-Verifica se os inputs são números inteiros (sao_inteiros, coord_corretas)

-Passa as coordenadas para tuplos (coordenadas_para_tuplo)

-Cria uma lista de tuplos com as coordenadas nos habitantes/poções, que verifica se existe sobreposição/repetição das mesmas (coord_lista)

-Através das coordenadas guardadas na lista, calcula os 'vizinhos' das mesmas (vizinhos)

-Cria um grafo (=dicionário) que contém os vizinhos das coordenadas dos habitantes

-De seguida, utilizando as funções das aulas teóricas, para 1 único habitante utilizámos a função (find_shortest_path) e para as restantes opções a função (find_all_paths)

-Verificam se existem coordenadas iguais em diferentes listas dependendo do número de habitantes –função das aulas teóricas - (tem_duplicados'P') *

*Sendo P o número de habitantes

Nota: A função find_shortest_path foi consultada no website:http://www.python.org/doc/essays/graphs/

Depois dos prints, inputs e verificações iniciais, na função Main, cria-se o tabuleiro/grelha preenchidas com 0's, o grafo dos vizinhos e ainda a lista das coordenadas em forma de tuplos.

De seguida, se for um unico habitante, utiliza-se a função find_shortest_path para calcular o caminho mais curto para a poção, e depois de encontrado substitui-se ,na grelha/tabuleiro, os 0's por 1's ao longo do caminho desde o inicio até à poção.

No caso de serem entre 2 a 7 habitantes,1º calculam-se todos os caminhos possiveis para cada habitante através da função find_all_paths que os coloca num dicionário (em que cada caminho é uma lista de tuplos) e, utilizando a função que verifica se existem coordenadas duplicadas nos diferentes caminhos (tem_duplicados'P'), eliminam-se os caminhos que as têm, ficando apenas com um caminho que para cada habitante, de modo a que não se cruzem. Já com cada habitante com o seu caminho, percorre-se o caminho de cada habitante na grelha/tabuleiro, substituindo na mesma(o) o 0 pelo correspondente número (2-7), fazendo ainda print de "TOUTATIS".

Por fim, se não existissem caminhos que não se cruzassem o programa faz print de "ALESIA".

Para o 2º método (ler ficheiro), utilizámos o método replace para apagar o caracter \n que cria os parágrafos mas não conseguimos apagar os 'white spaces' de modo que, não havendo espaços no ficheiro .txt o programa funciona, caso contrário dá erro ao tentar executar.