Projeto – Resolver uma sopa de letras!

Conjunto de dados:

* Lista de palavras (a procurar);
* Grelha de Letras (onde procurar).

Devolve:

* Lista de coordenadas;
* Direção;
* Devem ser assinaladas na janela de interação.

Ficheiro do Jogo:

* Nome do puzzle na 1º linha;
* A lista de palavras seguidas de uma vírgula;
* A Grelha de letras
* Ex ficheiro válido – capitais.txt

**Grelha de Letras:**

* Número fixo de linhas e colunas (*dimensões da grelha*);
* As dimensões são variáveis para os diferentes puzzles;
* As letras estão separadas por espaços;
* Posicionamento das letras:
  + Em Coordenadas (linha, coluna):
    - Linha (entre 0 e o (nº de linhas – 1));
    - Coluna (entre 0 e o (nº de colunas – 1));

**Janela de interação:**

* Onde será apresentado o puzzle e a respectiva resolução.
* Fornecida para o projecto.

**Tipos abstratos de informação**

**Tipo “*direcao*”:**

* Tipos:
  + ‘N’,’S’,’E’,’W’,’NE’,’NW’,’SE’,’SW’

1. **Reconhecedores:**
   * e\_direcao : universal -> lógico
     + e\_direcao(arg) tem o valor verdadeiro se o arg for do tipo *direcao e* falso caso **contrário**.
   * e\_norte : direcao -> lógico
     + e\_ norte(arg) tem o valor verdadeiro se o arg for o elemento ‘N’ e falso caso **contrário.**
   * e\_sul : direcao -> lógico
     + e\_ sul(arg) tem o valor verdadeiro se o arg for o elemento ‘S’ e falso caso **contrário.**
   * e\_leste : direcao -> lógico
     + e\_ sul(arg) tem o valor verdadeiro se o arg for o elemento ‘E’ e falso caso **contrário.**
   * e\_oeste : direcao -> lógico
     + e\_oeste(arg) tem o valor verdadeiro se o arg for o elemento ‘W’ e falso caso **contrário.**
   * e\_nordeste : direcao -> lógico
     + e\_ nordeste (arg) tem o valor verdadeiro se o arg for o elemento ‘NE’ e falso caso **contrário.**
   * e\_noroeste : direcao -> lógico
     + e\_ noroeste (arg) tem o valor verdadeiro se o arg for o elemento ‘NW’ e falso caso **contrário.**
   * e\_sudeste : direcao -> lógico
     + e\_ sudeste (arg) tem o valor verdadeiro se o arg for o elemento ‘SE’ e falso caso **contrário.**
   * e\_sudoeste : direcao -> lógico
     + e\_ sudoeste (arg) tem o valor verdadeiro se o arg for o elemento ‘SW’ e falso caso **contrário.**
2. **Testes:**
   * direcoes\_iguais : direcao x direcao -> lógico
   * direcoes\_iguais(d1,d2) devolve o valor verdadeiro se as direções d1 e d2 forem iguais e falso caso contrário.
3. **Outras Operações:**
   * direcao\_oposta: direcao -> direcao
   * direcao\_oposta(d) devolve a direção oposta de d de acordo com a rosa dos ventos
   * Exemplo: >>> direcao\_oposta(‘N’) , Devolve : ‘S’

**Tipo “*coordenada*”:**

* Constituido por dois inteiros e uma *direcao.*
* 2 Números indicam uma posição da grelha.

1. **Construtor:**
   * Coordenada : N0 x N0 x direcao -> coordenada

Coordenada(l,c,d) tem como valor a coordenada referente à posição (l,c) e direção d.

Em caso de erro, o construtor deve gerar um erro de valor (ValueError) com a seguinte mensagem: “coordenada: argumentos inválidos”

1. **Seletores:**

* Coord\_linha : coordenada -> N0
  + Coord\_linha(c) tem como valor a linha da coordenada.
* Coord\_coluna : coordenada -> N0
  + Coord\_coluna(c) tem como valor a coluna da coordenada.
* Coord\_direcao : coordenada -> N0
  + Coord\_direcao(c) tem como valor a linha da coordenada.

1. **Reconhecedor:**
   * e\_coordenada : universal -> lógico

e\_coordenada(arg) tem o valor verdadeiro se arg for do tipo coordenada e falso caso contrário.

1. **Testes:**
   * coordenadas\_iguais: coordenada x coordenada -> lógico

coordenadas\_iguais(c1,c2) devolve um valor verdadeiro se as coordenadas c1 e c2 forem iguais e falso caso contrário.

1. **Outras Operações:**
   * Coordenada\_string: coordenada -> string

Coordenada\_string(c) devolve a representação externa de c: uma cadeia de caracteres iniciada por parêntesis esquerdo ‘(’ seguido pelo número da linha e da coluna separados por vírgula e um espaço, seguido por parêntesis direito e traço ’)-‘ , após os quais se apresenta a direção

Ex:

>>> Coordenada\_string(coordenada(5,29,’NE’))

‘(5, 29)- NE’

**Tipo “*coordenada*”:**

* É uma matriz (m x n):
  + m número de linhas;
  + n número de colunas.

1. **Construtor:**
   * Grelha : lista de strings -> coordenada

Grelha(lst) tem como valor uma grelha (mxn) e a lista não pode ser vazia e todas as strings devem ter o mesmo comprimento.

Em caso de erro gerar um ValueError: “grelha: argumentos inválidos”

1. **Seletores:**
   * Grelha\_nr\_linhas : grelha -> N0

Grelha\_nr\_linhas(g) devolve o número de linhas da grelha g.

* + Grelha\_nr\_colunas : grelha -> N0

Grelha\_nr\_colunas(g) devolve o número de colunas da grelha g.

* + Grelha\_elemento: grelha x N0 x N0 -> carácter

Grelha\_elemento(g,l,c) devolve o carácter que está na posição (l,c) da grelha g.

No caso de (l,c) não ser uma posição válida mostrar ValueError: “grelha\_elemento: argumentos inválidos”

* + Grelha\_linha: grelha x coordenada -> string

Grelha\_linha(g,c) devolve a cadeia de caracteres que corresponde à linha definida segundo a direção dada pela coordenada c, r que inclui a posição dada pela mesma coordenada.