Exame, 09/01/2015 (Tempo: 3h)

INTELIGÊNCIA ARTIFICIAL

Identificação do aluno: _____ #: _____

Nome: _____

I

1. Analise as seguintes funções escritas em Python e explique o que fazem. Não precisa descrever o funcionamento interno das funções.

a)
```
def f(x,y,z):
    if y==[]:
        return [x]
    if z(x,y[0]):
        return [x]+y
    return [y[0]]+f(x,y[1:],z)
```

x é um elemento, y é uma lista e z é uma função de comparação. A função insere x, antes do elemento y ao qual z retornou verdadeiro

b)
```
def h(x,y,z):
    if x==[]:
        return y[:]
    if y==[]:
        return x[:]
    if z(x[0]) < z(y[0]):
        return [x[0]]+h(x[1:],y,z)
    return [y[0]]+h(x,y[1:],z)
```

A função compara por ordem o valor retornado por z aos valores das listas x e y. Se z(x[0]) for menor, este é adicionado à lista retornada chamando a função com os valores de x a seguir (semelhante para y). Quando uma das listas acaba, adiciona o resto da outra ao resultado.

2. Para efeitos de implementação de redes de Bayes em Python, as probabilidades condicionadas podem ser representadas como tuplos (*Var*,*Mothers*,*Prob*), em que *Var* é uma das variáveis da rede, *Mothers* é uma lista de tuplos representando uma das combinações possíveis de valores das variáveis mães de *Var*, e *Prob* é a probabilidade condicionada de *Var* dado *Mothers*. Programe uma função que, dada uma lista de tuplos representando todas as probabilidades condicionadas de uma rede, e dada ainda uma determinada variável da rede, retorna uma lista com todas as variáveis ascendentes dessa variável. Exemplo:

```
>>> bn = [ ( "C", [ ("A",True), ("B",True) ], 0.95), ( "C", [ ("A",True), ("B",False) ], 0.7), ( "C", [ ("A",False), ("B",True) ], 0.65), ( "C", [ ("A",False), ("B",False) ], 0.1), ( "D", [ ("C",True)], 0.77), ( "D", [ ("C",False)], 0.22), ( "B", [ ], 0.33) ]
>>> get_ancestors(bn, "D")
[ "A", "B", "C" ]
```

```python
def get_ancestors(bn, var):
    ancestors = []
    for tup in bn:
        if tup[0] == var:
            for mother in tup[1]:
                if mother[0] not in ancestors:
                    ancestors += get_ancestors(bn, mother[0])
                    ancestors.append(mother[0])
    return ancestors
```

II

1. Neste exercício, tem um conjunto de questões de escolha. Em cada alínea, apenas uma das opções dadas está certa, e apenas pode seleccionar uma delas. Cada resposta errada desconta 20% da cotação da alínea.

a) A frase "O pai do António é casado com a mãe da Teresa." pode ser representada em Lógica de Primeira Ordem da seguinte forma:

- ∃x Pai(x,Antonio) ∧ ∃y Mãe(y,Teresa) ∧ CasadoCom(Antonio,Teresa)
- ∃x Pai(x,Antonio) ∧ ∃y Mãe(y,Teresa) ∧ CasadoCom(x,y)
- ∃x ∃y Pai(x,Antonio) ∧ Mãe(y,Teresa) ∧ CasadoCom(Antonio,Teresa)
- ∀x ∃y Pai(x,Antonio) ∧ Mãe(y,Teresa) ∧ CasadoCom(x,y)
- Nenhuma das anteriores

[ ] [X] [ ] [ ] [ ]

b) Relativamente às redes de Bayes, indique a afirmação verdadeira

- Não permitem representar conhecimento impreciso
- São representadas por grafos não dirigidos
- Permitem representar as dependências entre as variáveis de um problema
- Permitem representar relações de herança entre entidades
- Nenhuma das anteriores

[ ] [ ] [X] [ ] [ ]

c) Relativamente às redes semânticas, indique a afirmação verdadeira

- Não é possível representar hierarquias de tipos
- Permitem representar conhecimento por omissão
- Nenhum tipo de rede semântica permite representar a negação ou a disjunção
- A rede semântica estudada na componente prática da disciplina é tão expressiva como a lógica de primeira ordem
- Nenhuma das anteriores

[ ] [X] [ ] [ ] [ ]

d) Uma consequência lógica do conjunto de fórmulas { A∨B, ¬B∨C∨D, ¬A , ¬D} é:

- ~B ∧C
- A ∨ D ∨ C
- A
- A ∨ ~B
- Nenhuma das anteriores

[ ] [X] [ ] [ ] [X]

e) O algoritmo de pesquisa em grafo (*graph-search*) difere do algoritmo de pesquisa em árvore (*tree-search*) em que:

A pesquisa em grafo utiliza-se transições sempre na forma de operadores STRIPS	
A pesquisa em grafo trabalha com um grafo de restrições	
A pesquisa em grafo não cria nós com estados repetidos no caminho de cada nó até à raiz	X
A pesquisa em grafo é completa e óptima	
Nenhuma das anteriores	

2. Está a ser desenvolvido um novo robô aspirador, sendo necessário implementar o respectivo algoritmo de controlo. Um pressuposto do algoritmo é que o espaço a limpar está organizado na forma de uma grelha de células quadradas. O aspirador consegue determinar se existe lixo para aspirar na célula em que ele está, bem como nas células vizinhas (*aqui, em frente, à esquerda, à direita* e *atrás*). As acções que o aspirador consegue executar são: aspirar o lixo na célula actual; mover-se para a célula em frente; e rodar 90º para a direita. Sempre que detecta lixo na vizinhança, o robô move-se para a célula em que está o lixo. No caso de não detectar lixo, o aspirador move-se para a célula em frente caso o tempo actual (em segundos) seja par, ou roda 90º para a direita caso contrário. Não precisa de preocupar-se com obstáculos, já que a acção de mover é automaticamente omitida caso haja um obstáculo em frente.

a) Identifique e caracterize as várias condições (proposições/predicados) que podem ser usadas para descrever as situações em que se pode encontrar o robô aspirador.

Lixo_em(c), c ∈ {aqui, em frente, à esquerda, à direita}

tempo_par

ou

detecta(x)

x ∈ {aqui, frente, direita, trás, esquerda}

Lixo_em(aqui)
Lixo_em(frente)
Lixo_em(esquerda)
Lixo_em(direita)

tempo(x)

x é par ou ímpar

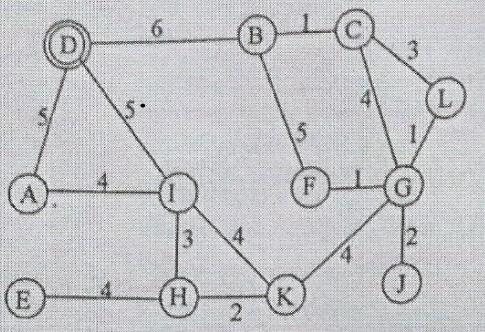
b) Especifique um conjunto de regras situação-acção que definam um comportamento adequado do robô aspirador. Pode fazê-lo na forma de uma tabela com as seguintes colunas:

- Situação - uma conjunção de condições
- Actualização - actualização das variáveis de estado, caso existam
- Acção - acção a executar pelo agente na situação indicada

Situação	Atualização	Acção
detetaLixo ∧ Lixo_em(aqui)	—	Aspirar
detetaLixo ∧ Lixo_em(frente)	—	Mover em frente
detetaLixo ∧ Lixo_em(direito)	—	Rodar 90° direita
detetaLixo ∧ Lixo_em(esquerda)	—	Roda 90° direita
¬ Lixo_em(c) ∧ tempo_par	—	Mover em frente
¬ Lixo_em(c) ∧ ¬ tempo_par	—	Rodar 90° direito

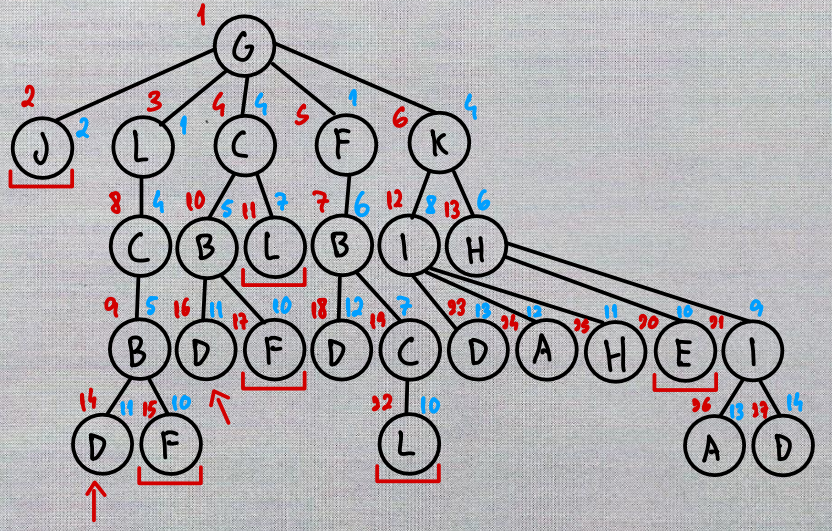
situação	atualização	acção
dalela (aqui)	———	aspira ()
dalela (frente)	———	avança () ∧ aspira ()
dalela (direita)	———	roda (90°) ∧ avança () ∧ aspira ()
dalela (trás)	———	roda (90°) ∧ roda (90°) ∧ avança () ∧ aspira ()
dalela (esquerda)	———	roda (90°) ∧ roda (90°) ∧ roda (90°) ∧ avança () ∧ aspira ()
¬ dalela (x) ∧ tempo (fim)	———	avança ()
¬ dalela (x) ∧ tempo (fim)	———	roda (90°)

3. O grafo a seguir apresentado representa um espaço de estados num problema de pesquisa, sendo D o estado objectivo (solução). Os custos das transições estão anotados junto às ligações do grafo.



a) Tomando o estado G como estado inicial, apresente a árvore de pesquisa gerada quando se realiza uma pesquisa de custo uniforme. Esta pesquisa é feita sem repetição de estados no caminho de qualquer nó até à raiz da árvore. Numere os nós pela ordem em que são acrescentados à árvore e anote também o valor da função de avaliação em cada nó. Em caso de empate nos valores da função de avaliação em dois ou mais nós, utilize a desempate com base na ordem alfabética dos respectivos estados.

ordem de adição
custo



b) Calcule o factor de ramificação médio da árvore gerada.

$$R = \frac{27 - 1}{13} = \frac{26}{13} = 2$$

N = 9
x = 3

c) Identifique semelhanças e diferenças entre a pesquisa em largura e a pesquisa de custo uniforme.

Ambos os algoritmos examinam os nós numa ordem definida e num grafo não ponderado ambos comportam-se da mesma maneira. Na pesquisa em largura os nós são ordenados de acordo com a sua distância à raiz enquanto que no outro é pelo custo acumulado. A pesquisa em largura não garante que o caminho encontrado é o mais curto pois não tem em atenção o custo. A pesquisa em largura pode ser usada quando queremos menor número de movimentos enquanto que o outro encontra o mais curto.

4. Considere o seguinte problema:
"*André, Bernardo e Cláudio dão um passeio de bicicleta. Cada um anda na bicicleta de um dos amigos e leva o chapéu de um dos outros. O que leva o chapéu de Cláudio anda na bicicleta de Bernardo. Que bicicleta e que chapéu levam cada um dos amigos?*"
Com vista à resolução do problema através de pesquisa com propagação de restrições, identifique as variáveis e respectivos valores possíveis, e represente a informação disponível através de um grafo de restrições.

(chapeu de , bicicleta de)

A ∈ {~~(B, C)~~, (C, B)}

B ∈ {(A, C), (C, A)}

C ∈ {(B, A), (A, B)}