Folha de exercícios 4

1. Escrever o predicado path com três parâmetros (i.e. com o terceiro parâmetro como uma lista com o caminho entre dois pontos) sem usar append.

2. Escrever quicksort usando diferença de listas.

3. O problema clássico *Torres de Hanoi* pode ser resolvido em Prolog com o seguinte predicado:

```
hanoi(1,A,B,C,[A para B]).

hanoi(N,A,B,C,Movs) :-
      M is  N - 1,
      hanoi(M,A,C,B,Ms1),
      hanoi(M,C,B,A,Ms2),
      append(Ms1,[A para B | Ms2],Movs).
```

   Escreva o mesmo predicado sem usar *append*.

4. Escrever os predicado preordem, inordem e posordem (da folha 3) sem fazer referência a append.

5. Reescreva o predicado flatten de forma que a lista resultado tenha os elementos da lista a ser tratada em ordem inversa.

6. Escrever o predicado inverse com diferença de listas.

7. Apresente um predicado que dada uma lista de listas de inteiros produza uma lista de somas dos inteiros. A lista resultado deve ser uma diferença de listas. Por exemplo para a query

$$?\mathsf{somalista}([[2,[1,3,1],3],[1,[5]],[5,7,1]],\mathsf{Soma}-[])$$

obtém-se como resultado $\mathsf{Soma} = [10, 6, 13]$.

8. As árvores binárias podem ser usadas para representar expressões aritméticas. Por exemplo a expressão $(c - (d/e))$ pode ser representada pela árvore $\mathsf{tree}(-, \mathsf{tree}(\mathsf{c}, \mathsf{nil}, \mathsf{nil}), \mathsf{tree}(/, \mathsf{tree}(\mathsf{d}, \mathsf{nil}, \mathsf{nil}), \mathsf{tree}(\mathsf{e}, \mathsf{nil}, \mathsf{nil})))$.

Parte do trabalho de um compilador é traduzir estas expressões em linguagem máquina. Supor que a linguagem de uma máquina para onde vamos gerar código tem um conjunto sequencial de registos numerados que podem conter resultados temporários. Supor também que a linguagem possui a instrução mov para colocar valores nestes registos e as instruções de aritmética básica add, sub, mul e div. Por exemplo, a expressão anterior seria traduzida na sequência:

[[mov, 1, c], [mov, 2, d], [mov, 3, e], [div, 2, 3], [sub, 1, 2]]

Defina nestes termos um predicado compile(Expr, Codigo).

9. Considere a relação pai/2.

a) Apresente um predicado que dado o nome de uma pessoa calcula o número de filhos dessa pessoa.

b) Escreva um predicado que produza uma lista de termos que relaciona todas as pessoas presentes na relação pai/2 com os seus filhos i.e. L = [p(joao, [maria, ana]), p(rui, [antonio, ze])].

10. Dado um mapa de ligações entre cidades (com informação sobre as distâncias e preço das ligações) pretende-se definir um predicado que encontre o melhor itinerário de ligação entre duas cidades. O mapa é definido pelas cláusulas arco(porto, lisboa, 300000, 5000$00).

Primeiro queremos encontrar um caminho óptimo (não necessáriamente o melhor de todos) em termos de distância. O método a usar pode ser do tipo (best first) que escolhe, entre um conjunto de ligações entre uma cidade e outras, a ligação que tem menor distância.

11. Um homem chega a uma paragem de autocarro às 9H00 de um dia e ai permanece 60 minutos (até às 9H59). Durante este período ele anotou o momento de chegada de cada autocarro que ai pára (regista o minuto). Os autocarros podem pertencer a carreiras diferentes e podem chegar no mesmo momento. No entanto autocarros da mesma carreira chegam sempre a intervalos regulares. Sabe-se ainda que todas as carreiras pararam nessa paragem pelo menos duas vezes.

Escreva um programa em Prolog que dada a lista ordenada com os minutos de paragem de cada autocarro observado, indica um conjunto de carreiras em que eles podem ser agrupados, identificando a respectiva frequência (i.e. o intervalo de tempo entre cada passagem). O resultado deve ser apresentado na forma de uma lista de pares em que o primeiro elemento representa o minuto da primeira paragem e o segundo a frequência de passagem.

Como exemplo a query

agrupa([0, 4, 5, 7, 10, 20, 20, 24, 30, 35, 37, 40, 44, 50, 50], Car)

Car = [(0, 10), (4, 20), (5, 15), (7, 30)]

o que significa que o primeiro autocarro parou ao minuto 0 e pára todos os 10 minutos, o segundo parou ao minuto 4 e pára todos os 20 minutos (parou aos 4, 24 e 44) etc etc.