Programmation Logique et Fonctionnelle - TP5
PROLOG

1 Arbres

Dans cet exercice, on utilisera des termes pour représenter des arbres binaires donts les noeuds sont étiquetés par des entiers :
- un arbre constitué d'un noeud racine étiqueté par `X`, d'un fils gauche `G`, et d'un fils droit `D` est noté `n(X,G,D)` ;
- une feuille est notée `f`.

Par exemple, `n(1,n(5,f,f),f)` est un arbre.

1.1

Spécifiez un prédicat `complet/2` tel que `complet(T,N)` soit vérifié si et seulement si `T` est un arbre complet de profondeur `N`. Un arbre est complet (pour mémoire) si et seulement si toutes ses feuilles sont à la même distance de la racine.

1.2

Spécifiez un prédicat `miroir/2` tel que `miroir(T1,T2)` soit vérifié si et seulement si `T1` est l'arbre symétrique de `T2`, i.e. la représentation graphique de l'un est identique à la reflexion dans un miroir de celle de l'autre.

1.3

Spécifiez un prédicat `construit/2` tel que `construit(T,N)` permette de construire un arbre `T` de pronfondeur `N` dont tous les noeuds sont étiquetés par des 0.

1.4

On appelle *arbre ordonné* un arbre tel que pour tout noeud `n(X,G,D)`, toutes les valeurs des noeuds de `G` sont au plus égales à `X` et toutes les valeurs des noeuds de `D` sont supérieures à `X`.
Spécifiez un prédicat `insere/3` tel que si `X` est un entier et `T` un arbre ordonné, alors `insere(X,T,R)` permette de construire un arbre ordonné `R` contenant les valeurs des noeuds de `T` et un nouveau noeud ayant la valeur `X`.

2 Représentation unaire des entiers

On représente les entiers par des termes de la manière suivante :
- 0 est représenté par 0 ;
- le successeur d'un entier `X` est représenté par `succ(X)`.

Par exemple, 3 est représenté par `succ(succ(succ(0)))`.

2.1

Spécifiez un prédicat `fois2/2` tel que `fois2(X,Y)` soit vérifié si et seulement si `Y` vaut `2X`.

2.2

Spécifiez un prédicat `somme/3` tel que `somme(X,Y,S)` soit vérifié si et seulement si `S = X+Y`.

2.3

Spécifiez un prédicat `produit/3` tel que `produit(X,Y,P)` soit vérifié si et seulement si `P = XY`.

2.4

Spécifiez un prédicat `factoriel/2` tel que `factoriel(X,Y)` soit vérifié si et seulement si `Y = X!`.

3 Approfondissement

On défini un graphe par des faits qui identifient les arcs.

```
arc(a,b).
arc(b,a).
arc(a,d).
arc(d,a).
arc(d,c).
arc(c,d).
arc(c,b).
arc(b,c).
arc(a,c).
arc(c,a).
```

On défini des couleurs par des faits.

```
couleur(rouge).
couleur(vert).
couleur(bleu).
```

On défini un coloriage comme une liste d'assignations de couleurs, comme par exemple :

```
[col(a, rouge), col(b, vert), col(c, bleu), col(d, vert)]
```

Spécifiez un prédicat `coloriage/2` tel que si `L` est une liste de sommets, alors `coloriage(L,C)` produit un coloriage de ces sommets tel que deux sommets reliés par un arc aient toujours des couleurs différentes. Vous pourrez spécifier un prédicat supplémentaire `compatible/3` tel que si `X` est un sommet, `C` une couleur et `W` un coloriage, alors `compatible(X,C,W)` est vérifié si et seulement si aucun sommet relié à `X` n'a la couleur `C` dans `W`.