Université Paris 7 - DEUG-MIAS Année 2004-2005

TP de *Logique et Circuits* n° 8

Calcul Propositionnel

On considère le type des formules du calcul propositionnel :

```
type conn_bi = Et | Ou | Impl
type formule = F of int | N of conn_bi * formule * formule | Non of formule;;
```

Une *formule* est une expression de type `formule`.

Une formule est dite *normale* si l'ensemble des étiquettes de ses feuilles est un segment initial d'entiers positifs.

Dans la suite, jusqu'à l'exercice 5 inclus, nous ne considérons que des formules normales.

Exercice 1 Écrire une fonction `arite` de type `formule -> int` qui renvoi le maximum de l'ensemble des étiquettes des feuilles de la formule.

Exercice 2 Écrire une fonction `eval` de type `formule -> bool list -> bool` qui s'applique à une formule `f` et à une liste `l` dont la longueur est au moins `(arite f)`, et qui renvoi la valeur de vérité de `f` relativement à l'interprétation qui associe à `F(n)` le `n`-ième élément de `l`.

Exercice 3 Écrire une fonction `eval_complet` de type `formule -> bool list` qui s'applique à une formule `f` et renvoi la liste des valeurs de vérité de `f` relativement à toutes les interprétations de ses feuilles (produites par la fonction `combi` de type `int -> bool list list` du TD 8).

Exercice 4 Écrire une fonction `valide` de type `formule -> bool` qui vérifie si une formule est valide (tautologique).

Exercice 5 Écrire une fonction `consequence_logique` de type `formule -> formule -> bool` qui vérifie si son premier argument est conséquence logique du deuxième.

Exercice 6 Écrire une fonction `normale` de type `formule -> bool` qui vérifie si son argument est une formule normale.

Exercice 7 Écrire une fonction `normalise` de type `formule -> formule` qui prend une formule et renvoi une formule normale équivalente.