UNIVERSITÉ DE STRASBOURG

**Département d'Informatique**
*LICENCE 2ème année*
*Travaux Pratiquess de Logique*
*Logique*

# TP2 : structures de données simples

## 1 Chaînes de caractères

Dans les versions "standard" de Prolog (Prolog d'Edimbourg), les chaînes de caractères ne constituent pas une structure de données à part entière : les chaînes de caractères sont des listes d'entiers (les codes ASCII des caractères). Les constantes chaînes de caractères sont représentées en tapant la suite des caractères entre cotes ("). Ainsi, `write("salut")` produit sur la sortie standard : `[ 115, 97, 108, 75, 74 ]`. Le prédicat `name/2` transforme un atome en chaîne de caractères et inversement.

### 1.1 Traitement des chaînes

On commence par définir quelques prédicats sur les chaînes de caractères.

- Écrire les clauses définissant un prédicat de nom `epelle/0` lisant un atome sur l'entrée standard et en écrivant ses lettres une par ligne.
- Définir le prédicat `aconcat/3` réalisant la concaténation de deux atomes. Ce prédicat est-il réversible ? Á quoi peut-il servir d'autre ?
- Définir le prédicat `substring/4` tel que `substring(Chaine, Debut, Nombre, Extrait)` réussit si `Extrait` est la sous-chaîne de `Chaine` commençant au caractère numéro `Debut` et comportant `Nombre` caractères. En déduire une définition Prolog de `subatom/4`, version des `substring` pour les atomes.
- Définir un prédicat de nom `palin/1` testant si son argument (qui doit être un atome) est un palindrôme.

### 1.2 Mutants

On considère la base de faits Prolog suivante :

```
animal(hibou).       animal(caribou).     animal(bouquetin).
animal(ours).        animal(tigre).       animal(alligator).
animal(grenouille).  animal(singe).       animal(hippopotame).
```

etc. avec ouistiti, crabe, becasse, tortue, vache, chevre, cheval, lapin, pelican, marmotte, canard, lama, tetard, pintade, truite, et d'autres.

Un mutant de première génération est un animal dont le nom (un atome) commence par le nom d'un animal, finit par le nom d'un autre tel que ce dernier commence comme finit le premier ... Ça n'est pas clair ? Exemple `caribouquetin` est un mutant descendant d'un `caribou` et d'un .....

Écrire un prédicat de nom `mutant/1` réussissant si son argument est un mutant de première génération (un vrai ...). Se servir de ce prédicat pour afficher tous les mutants de première génération.

Écrire un prédicat de nom `gdmutant/2` tel que `gdmutant(M, Gen)` réussisse en instanciant `M` avec un mutant de `Gen`ème génération. Remarque : une mutant de génération 0 est un animal dont le nom est présent dans la base de fait ; Un mutant de génération $n$ est le fils d'un mutant de génération $n_1$ et d'un mutant de génération $n_2$ tel que $n = n_1 + n_2$.

Généraliser avec le prédicat de nom `gdmutant/1`, cherchant les mutants de génération quelconque.

# 2 Arbres

## 2.1 Arbres binaires

Dans cette partie, nous considérons des arbres binaires d'entiers (par exemple) et les opérations associées en Prolog. Les arbres que nous considérons sont soit l'arbre vide (représenté, pour ce TP, par l'atome `nil`), soit un arbre de racine `X` et de sous-arbres gauche et droit respectivement `AG` et `AD` (représenté par le terme Prolog `t(AG, X, AD)`. Quel terme Prolog correspond à l'arbre donné (horizontalement) ci-dessous ?

```
                    nil
              3
                    nil
        7
                    nil
              2
                    nil
4
              nil
        2
                    nil
              8
                    nil
```

Définir, en Prolog les prédicats suivants :

| | |
|---|---|
| `estarbin/1` | teste si son argument est un arbre binaire |
| `ag/2` | extrait le sous-arbre gauche |
| `ad/2` | extrait le sous-arbre droit |
| `rac/2` | extrait la racine |
| `ed/2` | extrait l'extrémité droite |
| `eg/2` | extrait l'extrémité gauche |
| `dans/2` | teste si un entier est contenu dans un arbre |
| `nn/2` | donne le nombre de noeuds de l'arbre |
| `hauteur/2` | donne la hauteur de l'arbre |
| `affiche/2` | affiche, horizontalement, l'arbre binaire donné en argument |

Pour chacun des prédicats que vous aurez définis, posez-vous la question de sa réversibilité.