

## TD1 - Introduction

### Objectifs

- Introduction à la théorie des langages
  - Grammaire
  - Dérivation de mots
- Décomposition d'un problème en analyse lexicale et syntaxique

### Rappels et notations

On rappelle que le mot langage s'écrit sans u en français.

#### Exercice 1. Dérivation de règles

En théorie des langages, quand les symboles sont déjà des suites de symboles, on parlera de vocabulaire plutôt que d'alphabet et de phrases plutôt que de mots.

On considère le langage naturel (simplifié) suivant :

- Le vocabulaire  $A = \{\text{le}, \text{la}, \text{fille}, \text{raisin}, \text{cueille}\}$ .
- Une phrase est de la forme groupe nominal verbe groupe nominal.
- Un groupe nominal est de la forme déterminant nom.
- Les déterminants sont **la** ou **le**.
- Les noms sont **fille** ou **raisin**.
- Le verbe est **cueille**.

**Question 1.** Etablir la grammaire de ce langage.

**Question 2.** Montrer par dérivation que **la fille cueille le raisin** est une phrase du langage.

**Question 3.** Montrer que **le raisin cueille la fille** est une phrase du langage en construisant l'arbre de dérivation correspondant.

**Question 4.** Que pensez-vous de la deuxième phrase ?

**Question 5.** Existe-t-il des phrases grammaticalement incorrectes dans le langage ? Comment corriger la grammaire pour les interdire ?

**Question 6.** Combien de phrases différentes contient ce langage ?

## Exercice 2. Analyse lexicale et syntaxique

On considère le mini-langage de programmation suivant :

1. L'alphabet  $Alp$  du langage  $L$  est formé des lettres  $\{a, \dots, z, A, \dots, Z\}$ , des chiffres  $\{0, \dots, 9\}$ , du point  $\{.\}$ , de la quote  $\{'\}$  et des caractères  $\{+, -, *, /, =, &\}$ .
2. Un identificateur est composé de lettres et de chiffres et commence par une lettre.
3. Un nombre peut être un entier (composé de chiffres) ou un décimal (composé d'une partie entière et d'une partie fractionnaire séparées par un point).
4. Un opérateur est l'une des 4 opérations arithmétiques.
5. Une expression numérique est une suite de nombres ou d'identificateurs reliés par des opérateurs.
6. Une affectation numérique associe un identificateur à une expression numérique par le symbole égal ( $=$ ).
7. Un littéral est une suite de symboles entre simple quotes.
8. Une expression alphanumérique est une suite de littéraux ou d'identificateurs reliés par l'opérateur de concaténation (symbole  $\&$ ).
9. Une affectation alphanumérique associe un identificateur à une expression alphanumérique par le symbole égal ( $=$ ).
10. Une expression syntaxiquement correcte du langage défini sur  $Alp$  est une affectation numérique ou alphanumérique.

On cherche les expressions syntaxiquement correctes.

**Question 1.** Décomposer ce problème en analyse lexicale et analyse syntaxique.

1. Parmi les termes mentionnés, lequels correspondent à des lexèmes ?
2. Pour chaque spécification, relève-t-elle de l'analyse lexicale ou syntaxique ?
3. (Plus dur) Quelles grammaires peuvent correspondre chacune de ces analyses ?

### Exercice 3. Le langage SQL

Soit le sous-langage de SQL, permettant d'écrire des requêtes simplifiées de projection et de sélection sur une seule table ayant des champs de type chaîne et/ou de type entier :

**SELECT champ<sub>1</sub>, … , champ<sub>n</sub> FROM table WHERE condition**

où condition peut prendre l'une des formes suivantes :

- **champ<sub>i</sub> comp constante**
- **(condition oper condition).**

avec

- **oper** ∈ {OR, AND}
- **comp** ∈ {=, >, <}
- **constante** ∈ {'chaine', entier} où **chaine** est une suite finie de lettres ou de chiffres et **entier** une suite finie de chiffres.
- **champ<sub>i</sub>** et **table** sont des identificateurs (un identificateur étant composé de lettres et de chiffres et commençant par une lettre).

**Question 1.** Nous nous plaçons ici dans le cadre de l'analyse lexicale permettant de reconnaître les lexèmes utilisés dans une requête SQL. Donner une grammaire permettant d'identifier les différents lexèmes nécessaires pour les requêtes de ce langage SQL.

**Question 2.** Cette fois-ci, nous nous allons procéder à l'analyse syntaxique d'une requête SQL. En fonction des lexèmes obtenus lors de la question précédente, proposer une grammaire hors-contexte qui reconnaît les requêtes syntaxiquement bien formées.

**Les lexèmes obtenus précédemment deviennent donc les éléments terminaux de cette grammaire.**

### Exercice 4. (Pour s'entraîner) Langage binaires

Soit  $A = \{a, b\}$  un alphabet binaire.

**Question 1.** Proposer une grammaire formelle définissant chacun des langages suivants :

1. Le langage des mots de taille 6 :  $A^6$
2. Le langage de tous les mots :  $A^*$
3. Le langage des mots commençant par le symbole  $a$  et finissant par le symbole  $b$  :  $\{a \cdot u \cdot b \mid u \in A^*\}$
4. Le langage des palindromes :  $\{\varepsilon, a, b, aa, bb, aba, bab, \dots\}$ .
5. Le langage des mots tels que le symbole  $a$  n'est jamais suivi d'un autre symbole  $a$  :  
 $\{\varepsilon, a, ab, ba, bb, aba, abb, bab, bba, bbb, \dots\}$
6. L'intersection des deux langages précédents
7. Le langage des mots contenant autant de  $a$  que de  $b$