# Projet Nanoml (partie I) : l'arbre nano dom

## Description du projet

Le but du projet est de développer en C des outils autour des arbres N-aires étiquetés :

**Attention** : la saisie/la lecture depuis un fichier de ce type d'arbre n'est pas à faire dans le projet. Il fera l'objet d'un projet au second semestre.

## Travail à réaliser

### Bibliothèque pour affichage textuel enrichi

Pour ce projet, nous considèrerons que le texte enrichi est un texte écrit sur **80** caractères de large. Vous fournirez des outils pour **écrire** des mots dans le texte enrichi. Si la place nécessaire pour écrire le mot dépasse de la ligne. Le texte entame une nouvelle ligne pour écrire ce mot. Les mots peuvent s'écrire avec **3 modes** d'écriture :

* **NORMAL** : la casse des caractères respecte celle du mot écrit ;
* **MAJUSCULE** : toutes les lettres du mot sont écrites en majuscule indépendamment de la casse originelle ;
* **MINUSCULE** : toutes les lettres du mot sont écrites en minuscule indépendamment de la casse originelle ;

On pourra **changer de mode** d'écrire à n'importe quel moment dans le flux d'écriture.

On pourra **ouvrir des blocs** dans le texte. Cette opération consiste à placer le texte suivant à l'intérieur d'une "boite" délimité par des '+', '-' et des '|'.

#### Exemple 1 :

+------------------------------------------------------------------------------+

|toto |

+------------------------------------------------------------------------------+

Bien sûr, on pourra imaginer placer **des boîtes dans des boîtes** de manière récursive[[1]](#footnote-1). On pourra également **indenter** et **désindenter** le texte : Ajouter et retirer des décalages à droite au début de la ligne. L'utilisateur de la bibliothèque pourra **placer des puces** dans le texte au début d'une ligne. Le choix de la puce est indiqué par le **niveau d'indentation** du texte. On pourra aussi placer des boîtes indentées dans le texte et indenter le texte à l'intérieur d'une boîte.

#### Exemple 2 :

+------------------------------------------------------------------------------+

|+----------------------------------------------------------------------------+|

|| toto ||

|+----------------------------------------------------------------------------+|

|° TITI |

|° Pipo |

| \* Voila, c’est la fin. |

+------------------------------------------------------------------------------+

#### Remarque

Vous n’avez pas à vous soucier d’emboitement entre les indentations est les boites[[2]](#footnote-2). Ce contrôle sera réalisé au niveau de l’arbre nanodom.

### Bibliothèque pour arbres n-aires étiquetés

Les nœuds de cette structure d'arbre permettent de stocker des informations :

* D'une **étiquette** sous la forme d'un jeton (token) ;
* D'un **contenu** sous la forme d'une « chaine de caractères » ou d’un pointeur NULL ;
* Des **informations de navigation** dans l'arbre sous la forme des pointeurs :
  + Le père d'un nœud ;
  + Le premier fils ;
  + Le dernier fils ;
  + Le grand frère ;
  + Le petit frère.

Cette bibliothèque définir le type énuméré t\_token qui gère les **étiquettes** de l'arbre. Les étiquettes permettront de guider la fonction d'affichage élaboré exploitant la première partie.

Voici les valeurs que peut prendre le type t\_token :

* DOCUMENT : unique dans l'arbre il sera utilisé pour l'étiquette de la racine.
* SECTION : délimite une partie d'un document ou d'une section.
* TITRE : un nœud titre contiendra uniquement des nœuds MOT.
* LISTE : contient des nœuds MOT, des nœuds IMPORTANT et des nœuds ITEM.
* ITEM : contient des nœuds MOT, des nœuds IMPORTANT et des nœuds LISTE.
* IMPORTANT : contient des nœuds MOT.
* RETOUR\_A\_LA\_LIGNE : est une feuille de l'arbre.
* MOT : est une feuille de l'arbre.

Dans un nœud si l\_etiquette vaut MOT alors le champ le\_contenu pointera sur une chaine de caractères dans les autres cas, le champ le\_contenu sera NULL.

Vous placerez aussi des fonctions réalisant :

* Manipulation des nœuds
  + Création (allocation dynamique d’un nœud avec réglage des différents champs) ;
  + Destruction (libération d’un nœud **sans** prendre en compte la structure générale de l’arbre) ;
  + Modification (des fonctions pour modifier l’étiquette, le contenu ou une parenté d’un nœud **sans** prendre en compte la structure générale de l’arbre) ;
  + Déplacements : Ces fonctionnalités **prennent en compte la structure de l’arbre** pour tous les chainages restent cohérents.
    - Insertions : plusieurs fonctionnalités pour insérer un nœud supposément orphelin en **premier fils**, **dernier fils**, **après un frère**, **avant un frère**;
    - Extraction : une seule fonctionnalité pour sortir un nœud et tous ces descendants de l’arbre. Les frères et son père ne le reconnaissent plus.
* Affichage
  + Simple : représentant le contenu et la structure d’un nœud de l'arbre. Cette fonction doit vous aidez à débugger lors du développement. Elle affichera tout ce que vous jugerez utiles pour tracer arbre nanodom.
  + Élaboré : représentant le contenu textuel de l'arbre en exploitant la bibliothèque sur le texte enrichi.
    - Le document ouvre une boite pour y placer tous ces fils ;
    - Les sections ouvrent aussi des boites ;
    - Les titres s’écrivent en majuscules, commencent sur une nouvelle ligne et lorsque son contenu est écrit ils terminent la ligne en cours ;
    - Les listes augmentent l’indentation, commencent sur une nouvelle ligne et lorsque son contenu est écrit ils terminent la ligne en cours ;
    - Les items entament une nouvelle ligne et ils placent une puce en fonction du niveau d’indentation ;
    - Les éléments importants basculent l’affichage en majuscules pour son contenu (uniquement) ;
    - Les retours à la ligne forcent la terminaison de la ligne ;
    - Les mots affichent leurs champs contenus.
  + Enrichie : représentant la structure de l'arbre et son contenu avec en ajoutant des balises inspirées par le langage html et consorts. Les fils d’un élément T seront écrits par entre les balises <T> et </T>. L’élément « retour à la ligne » sera la balise « autofermante » : <RETOUR\_A\_LA\_LIGNE/>.
* Sauvegarde texte. Cette fonctionnalité agit comme l’affichage enrichie mais écrit les informations dans un fichier texte.

### Fonctions "main" pour lancer des tests sur le produit

* Tester l'affichage textuel : ce programme permet d'exécuter toutes les fonctions de la bibliothèque sur l'affichage du texte enrichi. Aucune mission particulière n'est attendue par ce programme. Vous le complétez au fur et à mesure de votre développement pour valider celui-ci.
* Tester l'affichage d'un arbre : programme construit un arbre puis l'affiche successivement avec la méthode simple puis la méthode élaborée.

### Fichier makefile

Ce fichier permet de réaliser les **dépendances** de vos fichiers pour la compilation de vos programmes. Vous utiliserez les options de compilation –Wall et –Wextra. Votre programme doit compilerez et **ne plus** provoquer d’avertissement.

## Rendu

Vous rendrez votre travail avant le 2 décembre 2022 à 23h59 sur icampus sous la forme d’une archive (zip ou rar). Votre archive devra contenir :

* Vos programmes C (bibliothèques (les fichiers .c et .h) et votre main) ;
* Votre makefile ;
* Un rapport présentant le projet (cf Kit de survie du rapport).

Votre rendu doit faire mention des deux membres du binôme.

**Attention** : si les deux membres du groupe font des rendus séparés. Le correcteur en corrigera un seul (au hasard). Pour être bien clair : Un ajout de dernière minute par l’un des membres du groupe pourra être corrigé et bénéficier au deux membres du groupe ou pourra être ignorer pour les deux selon l’exemplaire pioché par le correcteur.

**Attention** : les rendus (dépôt sur icampus) en retard seront pénalisés en appliquant la procédure stipulée par le règlement.

## Annexes

### Exemple de structure d’arbre nanodom

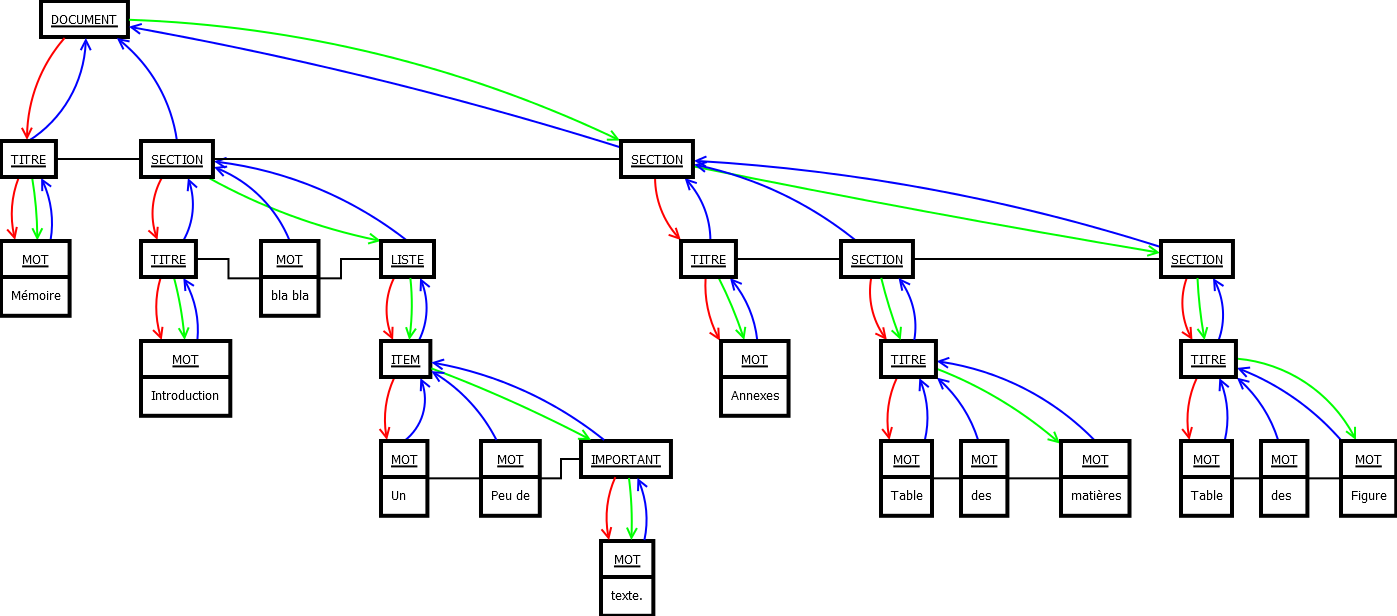


Figure . Exemple de début d’arbre. Les arcs rouges représentent des pointeurs allant d’un nœud vers son **premier fils**. Les arcs verts représentent les pointeurs allant d’un nœud à son **dernier fils**. Les arcs bleus représentent les pointeurs allant d’un nœud à **son père** les arcs noirs représentent **deux pointeurs** (l’un allant d’un nœud à son **petit frère** l’autre allant d’un **frère à son ainé**). Aucun pointeur vers NULL n’est représenté sur la figure par souci de lisibilité.

### Test affichage

Le document trace\_execution.txt reprend des affichages d’un arbre.

* Debugge tous les nœuds de l’arbre au moment de la construction des nœuds ;
* Debugge tous les nœuds de l’arbre après l’assemblage de la structure ;
* Affichage de la structure sous la forme élaborée ;
* Affichage de la structure sous la forme d’un texte enrichi ;
* Affichage des nœuds détruit lors de la destruction complète de l’arbre nanodom.

### Barème

Le tableau suivant présente une grille d’évaluation. Les cases grisées indiquent des sommes partielles des coefficients de la partie. Ce barème a pour but de vous aider à amorcer le travail et identifiant des points clés du projet.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Projet | Rendu conforme | | Makefile | 0,4 |
| Archive | 0,2 |
| Compilation | 0,4 |
| **1** | |
| Rapport | Document | Langue | 0,5 |
| Traitement de texte | 0,5 |
| **1** | |
| Introduction | Problème | 1 |
| Equipe | 0,5 |
| Méthode de travail | 0,5 |
| **2** | |
| Solution | Technologie | 1,5 |
| Difficultés surmontées | 0,5 |
| **2** | |
| Conclusion | Expérience | 0,5 |
| Prolongation | 0,5 |
| **1** | |
| **6** | | |
| Développement | Texte enrichi | Bibliothèque séparée | 0,5 |
| Gestion des variables | 1 |
| Ergonomie | 0,5 |
| **2** | |
| Arbre nanodom | Type token | 0,5 |
| Type nœud | 1 |
| Constructeur | 1 |
| Destructeur | 1 |
| Modificateurs | 1 |
| Insertions | 1,5 |
| Extraction | 1 |
| Affichage de débuggage | 1 |
| Affichage élaboré | 1,5 |
| Affichage enrichie | 1 |
| Sauve | 0,5 |
| **11** | |
| **13** | | |
| **20** | | | |

1. Il s'agit d'un concept récursif mais la programmation n'a pas besoin d'être récursif. [↑](#footnote-ref-1)
2. Pas de vérification que les indentations d’une boite sont finies avant de fermer cette boite. Pas de vérification que les boites à un niveau d’indentation sont fermées avant de désindenter le texte. [↑](#footnote-ref-2)