

# AAP - Fil Rouge : Compte rendu

AAPproximatif

Maistret James, Thieffry Émile, Chevalier Romain, Feng Yanli & Hong Yutong

8 janvier 2022

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Programme 1 : displayAVL.exe</b>	<b>2</b>
2.1	Développement . . . . .	2
2.1.1	Fonction de base pour créer l'arbre . . . . .	2
2.1.2	Lecture du fichier . . . . .	2
2.1.3	Affichage avec graphviz . . . . .	2
2.2	Jeux de test, exemples d'exécution . . . . .	3
<b>3</b>	<b>Programme 2 : indexation.exe</b>	<b>3</b>
3.1	Développement . . . . .	3
3.1.1	Fonction de base pour créer l'arbre . . . . .	3
3.1.2	Calcul des paramètres de l'arbre . . . . .	3
3.1.3	Recherche de mots . . . . .	3
3.2	Jeux de test, exemples d'exécution . . . . .	3
<b>4</b>	<b>Programme 3 : anagrammes.exe</b>	<b>3</b>
4.1	Développement . . . . .	3
4.2	Jeux de test, exemples d'exécution . . . . .	3
<b>5</b>	<b>Gestion de projet</b>	<b>3</b>
<b>6</b>	<b>Conclusion</b>	<b>3</b>

## Table des figures

1	Rotation simple à gauche . . . . .	2
4	Restructuration des mailles pour le second programme . . . . .	3
2	Construction de l'arbre pour les 10 premiers prénoms de <b>PrenomsV2.txt</b> . . . . .	4
3	Exemple arbre avec les 20 premiers prénoms de <b>PrenomsV1.txt</b> . . . . .	4

# 1 Introduction

Création d'arbre équilibrés en langage C, pour le fil rouge d'APP de l'année 21-22. Mise en place de trois programmes, le premier pour créer une image d'un arbre AVL à partir d'une liste de mot, le second pour indexer un dictionnaire dans un arbre AVL dont le tri des mots est basé sur leur signature et le dernier programme, qui repose sur le second, permet de rechercher les anagrammes d'un mot.

## 2 Programme 1 : displayAVL.exe

### 2.1 Développement

#### 2.1.1 Fonction de base pour créer l'arbre

**Rotations simples** Le programme de la rotation simple à droite était donné pour la rotation de gauche, nous avons raisonné par analogie et le calcul des facteur de déséquilibre est donné ci-dessous. En se basant sur la figure 1, on a  $a = h(A_g) - h(A_d)$ ,  $b = h(B_g) - h(A)$  et  $h(A) = 1 + \max(h(A_g), h(A_d))$  où  $h$  est la hauteur du noeud. Alors

$$\begin{aligned}
 b' &= h(B_g) - h(A_g) \\
 &= b + h(A) - h(A_g) && \text{en utilisant la définition de } b \\
 &= b + 1 + \max(h(A_g), h(A_d)) - h(A_g) && \text{en utilisant la définition de } h(A) \\
 &= b + 1 + \max(0, h(A_d) - h(A_g)) \\
 &= b + 1 + \max(0, -a) && \text{en utilisant la définition de } a \\
 &= 1 + b - \min(a, 0)
 \end{aligned}$$

Et de même,  $a' = 1 + a + \max(b', 0)$

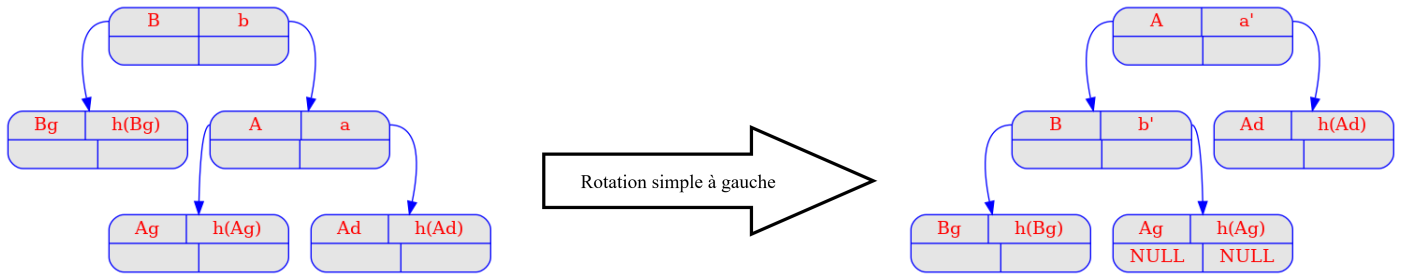


FIGURE 1 – Rotation simple à gauche

**BalanceAVL** Algorithme donné pour la première partie de la fonction, la seconde partie est l'exacte analogie pour un arbre qui penche de l'autre côté. Lors du développement de cette fonction, nous nous sommes heurté à un problème pour les tests afin de savoir l'arbre penche, en effet pour faire une rotation double, il faut que le facteur de déséquilibre soit forcément de 2 en valeur absolue et pas seulement non nul, ie l'arbre est bien déséquilibré. Mais une fois rentré dans la rotation double, pour savoir si on fait une rotation double du même côté, ou des deux côtés différents il suffit que l'arbre penche, ie que le facteur de déséquilibre du second nœud à tourner soit non nul.

**InsertAVL** Algorithme donné donc pas de remarques particulières.

#### 2.1.2 Lecture du fichier

En se basant sur [1], nous lisons ligne par ligne le fichier, car il y a mot par ligne dans le fichier, en ajoutant un compteur de lignes lues afin de respecter le nombre de mots à mettre dans l'arbre renseigné par l'utilisateur.

#### 2.1.3 Affichage avec graphviz

Les fonctions permettant de générer le fichier .dot étaient déjà données, néanmoins le programme ne prenait en compte les noms composés (ie les noms avec un tiret dedans). Afin de corriger cette erreur, il a fallu ajouter des guillemets (donc la séquence suivant \"%s\") dans les noms des blocs graphviz à afficher. De plus, l'affichage du facteur de déséquilibre n'était pas configuré, nous l'avons donc ajouté à côté du nom de famille du nœud.

## 2.2 Jeux de test, exemples d'exécution

On trouve en figure 2 , les différentes étapes de construction de l'arbre pour 10 premiers mots du fichier `PrenomsV2.txt`.

## 3 Programme 2 : indexation.exe

### 3.1 Développement

#### 3.1.1 Fonction de base pour créer l'arbre

**Restructuration des mailles** Pour ce programme, nous avons rajouté un champ dans chaque maille de l'arbre. Dans un premier temps, nous avons décidé que le champ `T_avl NodeAVL->val` deviendrait la signature des mots contenus dans le nouveau champ `T_avl NodeAVL->list_mots` qui contient la liste de tout les mots comportant la même signature. Voir illustration des champs d'une maille en figure 4a et un exemple en figure 4b.

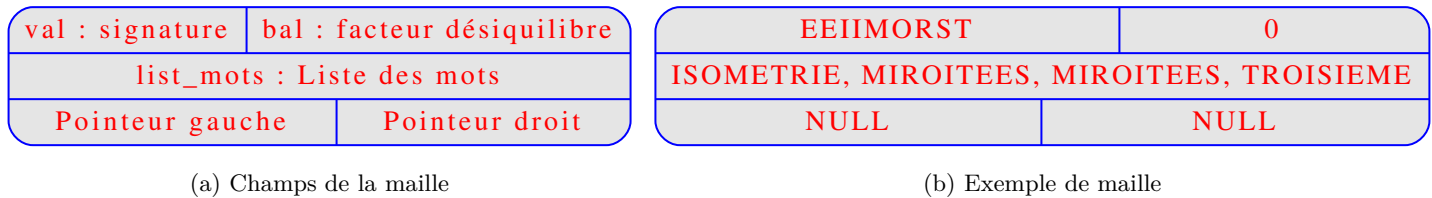


FIGURE 4 – Restructuration des mailles pour le second programme

### Ajout dans les mailles

#### 3.1.2 Calcul des paramètres de l'arbre

#### 3.1.3 Recherche de mots

## 3.2 Jeux de test, exemples d'exécution

## 4 Programme 3 : anagrammes.exe

### 4.1 Développement

### 4.2 Jeux de test, exemples d'exécution

## 5 Gestion de projet

## 6 Conclusion

## Références

- [1] Delftstack. Lire le fichier ligne par ligne en utilisant fscanf en c. <https://www.delftstack.com/fr/howto/c/fscanf-line-by-line-in-c/>, Mars 2021.
- [2] SARL Infini Software. Koor.fr. <https://koor.fr/C/Index.wp>, 2022.

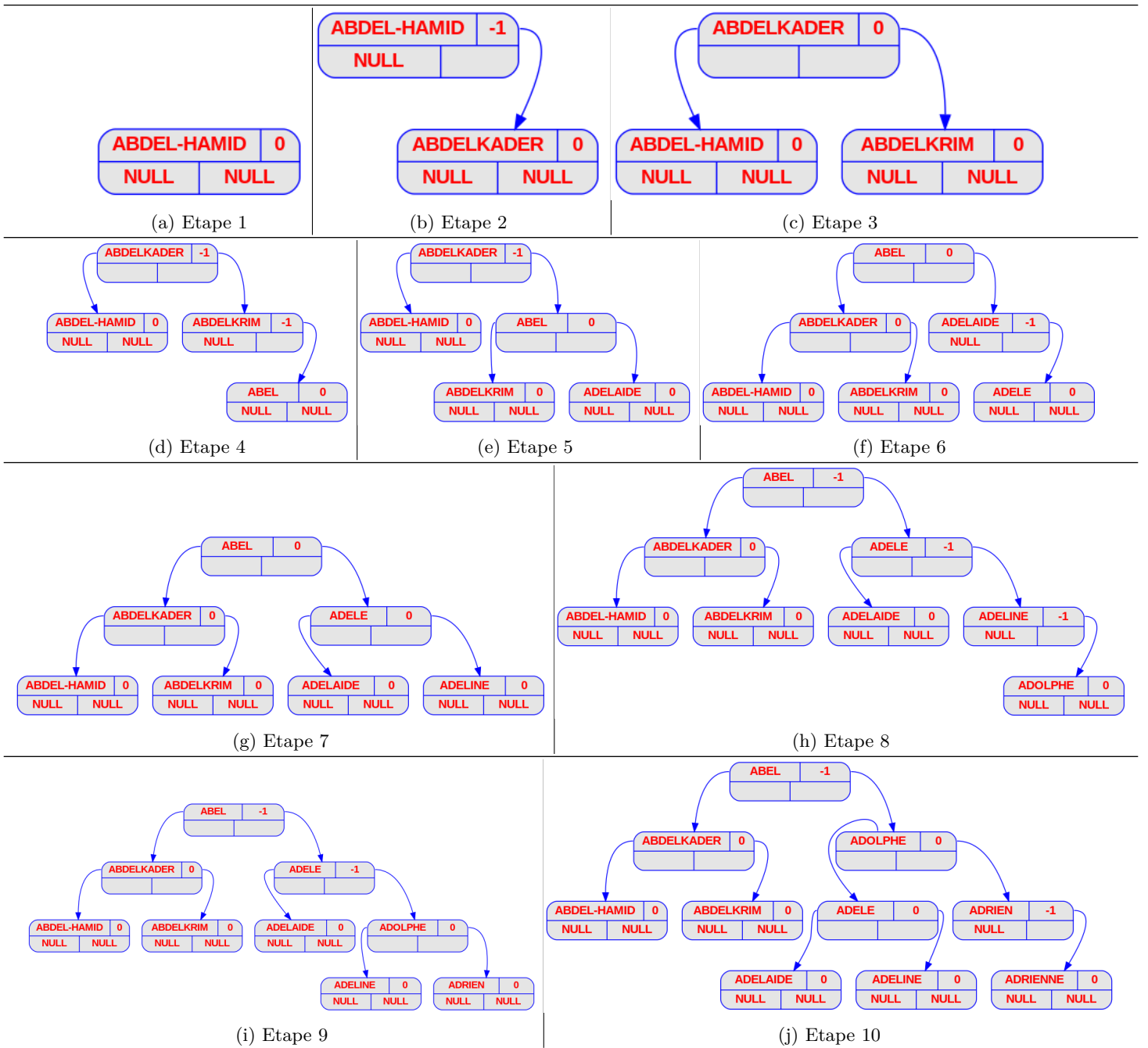


FIGURE 2 – Construction de l'arbre pour les 10 premiers prénoms de PrenomsV2.txt

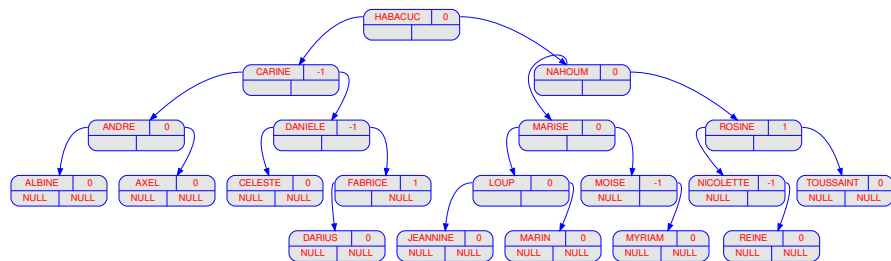


FIGURE 3 – Exemple arbre avec les 20 premiers prénoms de PrenomsV1.txt