,

Rapport Projet 3 — PIM

Auteurs :

VIGNAUX Adrien **BLANCHARD Enzo**

Professeur :

HAMROUNI Zouheir

Projet de première année de SN année : 2024-2025

Table des matières

1. Introduction 3
    1.1. Objectifs 3
    1.2. Description du problème 3
2. Contenu du projet 3
    2.1. Structure générale du projet 3
    2.2. Choix effectués 4
    2.3. Architecture de l'application modules 4
    2.4. Principaux algorithmes 4
    2.5. Démarches de test 4
3. Gestion du travail 4
    3.1. Organisation de l'équipe 4
    3.2. Difficultés rencontrées 4
4. Conclusion 5
    4.1. Bilan technique 5
    4.2. Bilan personnel et collectif 5
        4.2.1. De Adrien VIGNAUX 5
        4.2.2. De Enzo BLANCHARD 5

1. Introduction

Lors de ce module de Programmation Impérative, nous avons déjà pu réaliser deux mini-projets individuels : le premier étant sur la création du jeu des 13 allumettes, le second étant sur l'application des modules et de la généricité en lien avec des LCA (Listes Chaînées Associatives).

1.1. Objectifs

Pour ce troisième et ultime projet du module, nous sommes en binôme, et il nous est demandé d'écrire deux programmes :

- Le premier doit compresser des fichiers textes en utilisant le codage de Huffman ;
- Le second doit permettre de les décompresser.

Ce rapport débutera sur une présentation large du problème pour se poursuivre par nos démarches de résolution. Nous présenterons ensuite nos différents choix ainsi que les principaux rendus de ce projet (modules par exemple). Puis, nous expliquerons notre gestion du travail en équipe, avec les diverses difficultés rencontrées et ce qui pourrait être amélioré à l'avenir. Nous clôturerons ce rapport par un bilan technique, puis personnel.

1.2. Description du problème

Le codage de Huffman permet de compresser, sans perte, des données telles que des textes, des images ou encore du son. Nous sommes ici intéressés par la compression de texte.

Ce principe revient à compter le nombre d'occurrences de chaque caractère présent dans un texte, puis de leur inscrire une série de bits dont la taille dépendra de la fréquence d'apparition du caractère dans le texte. Ainsi, un caractère "commun" sera codé sur peu de bits, tandis qu'un caractère "peu commun" sera codé sur davantage de bits.

Assurément, nous devrons permettre la compression ainsi que la décompression du texte, sinon il nous serait impossible de récupérer et utiliser son contenu initial, avant sa compression.

Pour cette première version du rapport, nous resterons exclusivement sur la partie de compression de texte, le reste sera abordé ultérieurement.

2. Contenu du projet

2.1. Structure générale du projet

Notre projet se regroupe en plusieurs fichiers qui seront explicités au fur et à mesure de ce rapport. Voici la liste :

- Les fichiers arbre.adb et arbre.ads : module "Arbre" permettant de manipuler l'Arbre de Huffman ;
- Les fichiers table.adb et table.ads : module "Table" permettant de manipuler diverses tableaux spécifiques tel que la Table de Fréquence et la Table de Huffman ;
- Le fichier compresser.adb : premier fichier principal réalisant la compression d'un texte à partir du principe de codage de Huffman ;

- Le fichier decompresser.adb : second fichier principal réalisant la décompression de notre texte préalablement compressé ;
- Le fichier test_compresser.adb : fichier de réalisation de tests assurant le bon fonctionnement de compression du texte ;
- Le fichier test_decompresser.adb : fichier de réalisation de tests assurant le bon fonctionnement de décompression du texte.

2.2. Choix effectués

Nous avons choisi pour la structure de l'arbre, une liste chaînée car c'était la structure que l'on a le plus manipulé jusqu'ici.

Nous avons aussi choisi de mettre la capacité en paramètre de généricité pour pouvoir définir des tableaux de taille du nombre de caractère différents dans le texte.

2.3. Architecture de l'application modules

Nous avons choisi de faire deux modules. Un premier module qui gère tout les sous-programmes, lié à l'arbre. Un autre qui est générique pour pouvoir définir plusieurs tableaux différents.

2.4. Principaux algorithmes

- Tri_Table
- Ecrire_Octet
- Arbre_Huffman
- Creer_Table_Huffman
- Compresser_Texte

2.5. Démarches de test

3. Gestion du travail

3.1. Organisation de l'équipe

La répartition des tâches lors de la réalisation de ce projet s'est effectué comme telle :

3.2. Difficultés rencontrées

Durant la réalisation de ce projet, nous avons pu rencontrer certaines difficultés qui valent la peine d'être explicitées.

Premièrement, nous avons eu du mal à comprendre le fonctionnement de l'arbre de Huffman, ou plutôt comment elle s'implémente d'un point de vue algorithmique. Cela nous a demandé un peu plus de temps que prévu lors des prémices du projet, afin de partir sur de bonnes bases.

De plus, nous avions toujours en tête d'optimiser le programme dès que possible, cependant pour un projet qui commence à avoir une plus grande ampleur il est essentiel de débuter sur une base fonctionnelle (même si non optimisée), puis par la suite de l'améliorer du mieux que possible. Ainsi, nous partions souvent dans des idées bien trop complexes, qui n'aboutissaient pas forcément.

Enfin, il nous a été primordial de bien gérer notre temps, afin de pouvoir avancer efficacement dans la complétion du projet et de rendre notre travail dans les temps lorsque c'était demandé. C'était une difficulté continue qui ne devait être négligée à aucun moment, sous peine de retour de bâton imprévu.

4. Conclusion

4.1. Bilan technique

4.2. Bilan personnel et collectif

4.2.1. De Adrien VIGNAUX

Ce projet est très enrichissant au niveau du travail en équipe car c'est le premier projet long que l'on fait à deux.

4.2.2. De Enzo BLANCHARD

Finalement, ce projet était notre première approche du monde de l'ingénierie : entre la gestion d'équipe et du travail, les deadlines à respecter ou encore le processus de création, d'innovation pour un rendu encore et toujours plus performant, efficace, ce projet était assez complet et nous a permis de pleinement manipuler tous les aspects de la programmation impérative.