RAPPORT PROJET C-WIRE

Jérémie Konda, Maryam Lkhluf, Jibril Boucham MEF1 PréIng2

Ici se tient le rapport de notre projet C-Wire 2024-2025.

Le projet **C-Wire** consiste à concevoir un outil qui a pour but d'analyser et de trier les données de distribution de différentes stations. Afin de communiquer et être au plus efficace nous avons créé un groupe whatsApp en plus du dépôt gitHub où nous faisions chacun les commits de nos avancements. Nous avons chacun travailler de notre côté et nous nous réunissions pour tester nos essais sur les ordinateurs de l'école en plus des séances de TD.

1) Répartition des tâches

Ici un tableau qui répertorie les fonctions que chacun a conçu, lorsqu'une fonction apparaît pour deux personnes cela veut dire que nous sommes **deux à avoir travailler dessus.**

Les commentaires en gris expliquent à quoi nous servent les fonctions dans cette analyse.

Jérémie	Maryam	Jibril
Partie **Shell** validation_argument() afin que les entrées interdites tel que hvb indiv ou hva indiv ne soient entrées, que le minimum et maximum d'argument soient correct. preparer_dossier() afin de créer les dossiers requis. verifier_executable_c() vérifie et compile le programme C. filtrer_donnees() créer le fichier temporaire et filtre les données selon les arguments traitement_principal() constitue les fichiers de sorties avec les bonnes en-têtes selon les cas --- Partie **Documentation** Gestion du répertoire Github Réalisation du Read me partie C	Partie **Shell** verifier_installer_gnuplot() vérifie que gnuplot et installer, le cas échéant le demande à l'utilisateur. filtrer_donnees()créer le fichier temporaire et filtrer les données selon les arguments. traitement_principal()constitue les fichiers de sorties avec les bonnes en-têtes selon les cas. generer_lv_all_minmax() génère l'histogramme pour le cas où les arguments sont lv all. --- Partie **Documentation** Gestion du répertoire Github Réalisation du Read me partie Shell	Partie **C** ecrire_donnees() fonction qui contient une fonction récursive pour écrire les données sans les en-têtes. traiter_entree() traite ligne par ligne les entrées. **Création des fichiers .c et .c, création de la structure de l´avl ainsi que toutes les fonctions propres aux avl (équilibrage).** Makefile --- Partie **Documentation** Gestion du répertoire Github Réalisation du Rapport

Nous avons tous ciblé les points forts et points faibles de chacun afin de mieux répartir les tâches et ainsi être d'autant plus efficaces.

2) Planning de réalisation

Semaines	Réalisations
25/11	Prise de connaissance du projet, et de ses attendus.
02/12	Récapitulatif. Répartition des tâches. et début partie C et partie shell.
09/12	Création du répertoire github. Création des différents dossiers. Premiers dépôt de nos commits. Avancée Shell et C. Début Gnuplot.
16/12	Finalisation de Shell et C. Finalisation Gnuplot. Rédaction du Readme. Rédaction du Rapport. Relecture et ajustement. Livrable le 22/12.

(Veuillez tenir compte du fait que nous avons eu des examens les semaines du 25/12, 02/12 et 09/12.)

3) Limites de notre projet

- Les filtres et traitements actuels (`awk`, `grep`, `cut`, etc.) ne sont **pas optimisés pour des fichiers volumineux**. Dans le cadre de notre projet cela reste bien sûr fonctionnel, mais il peut y avoir des performances limitées.
- Nous avons dû être attentifs aux **fuites mémoires** en utilisant liberer_AVL().
- Entre le script Shell, le code C et les différents dossiers, il a fallu un certain temps pour structurer convenablement le projet. En plus de nos examens de fin d'années, nous avons dû faire preuve d'efficacité et d'organisation afin d'avancer au mieux

En conclusion, nous avons fait de notre mieux pour fournir un code robuste et lisible afin de répondre au cahier des charges. Les fonctionnalités demandées par ce dernier sont validées et fonctionnelles.