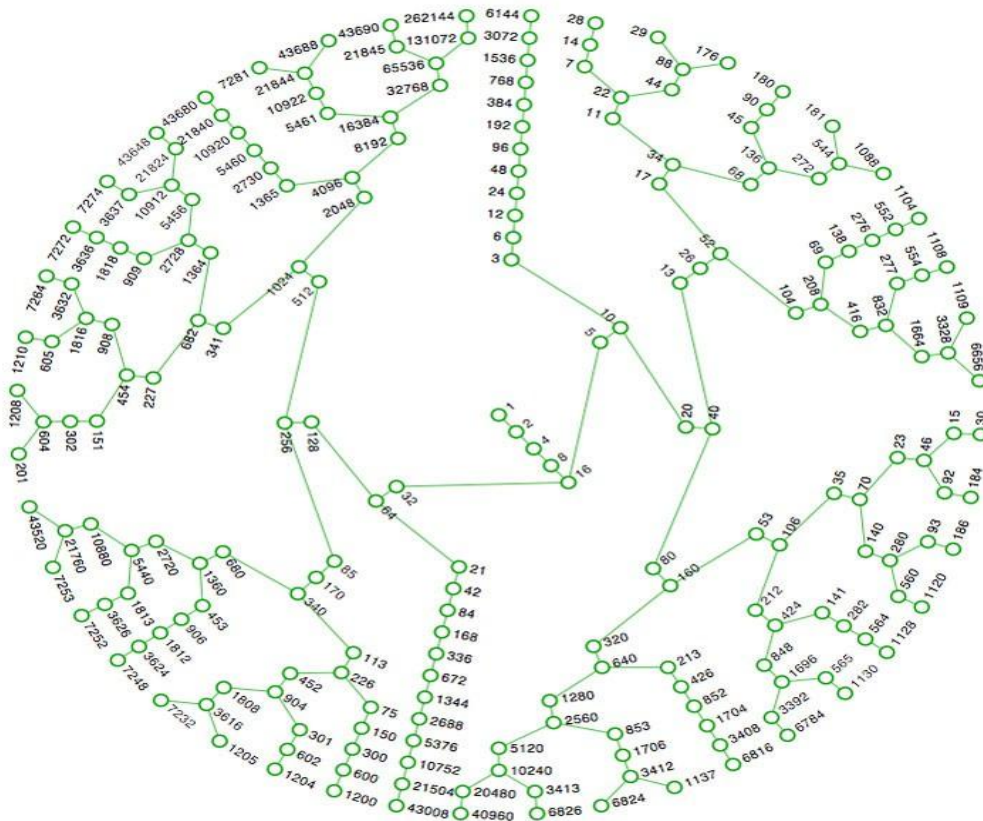


Script Shell

RAPPORT PROJET

COSTA Mathéo, AÏT CHADI Anissa

PréING2 MI Groupe 3



Introduction

L'objectif de ce projet est de réaliser, en groupe, un programme nécessitant de combiner langage C et surtout Shell tout en respectant un cahier des charges précis. Dans ce rapport, nous allons donc expliquer notre démarche ainsi que les principales parties du code avec des captures d'écran pour montrer le rendu sur la console. Nous allons ensuite parler des problèmes que nous avons rencontrés et conclure sur ce que nous a appris ce projet.

SOMMAIRE

I- Démarche suivie (p.4)

**II- Documentation & outils pour mettre en oeuvre notre projet
(p.6)**

III- Explication des principales fonctions et affichage console (p.7)

IV- Problèmes rencontrés (p.14)

V- Conclusion (p.15)

Démarche suivie :

Tout d’abord, expliquons comment s’est formé notre groupe.

Nous avons déjà eu à travailler ensemble, notamment pour le projet TIPE, le projet d’algo de l’an dernier ainsi que le Huffman, et de nombreux autres projets nécessitant un travail de groupe, ce qui fait que nous connaissions déjà la façon de travailler de l’autre, le binôme était donc une “évidence”.

À présent, nous allons voir comment nous nous sommes organisés pour mener à bien ce travail.

Lorsque nous avons vu le sujet, nous nous sommes concertés pour poser les objectifs à réaliser et pour faire une première ébauche de la répartition des tâches. Nous avons donné nos idées pour la réalisation du projet : quels outils nous allions avoir besoin, quelles devaient être les grandes bases pour le projet (deux nombres pris en compte, un calcul à réaliser, des graphiques à tracer et des fichiers à créer).

Ensuite, nous avons bien relu les indications du projet et essayé de réfléchir à comment le faire. Ainsi, nous avons utilisé, pour certaines parties, quelques lignes de pseudo code pour bien clarifier les idées avant de les traduire en C et de faire l’ébauche du script.

Nous communiquions principalement sur Snapchat afin d’avoir une communication plus rapide et pour pouvoir envoyer des photos, des vidéos ou même juste pour faire des messages vocaux afin de pouvoir les écouter et y répondre quand on le souhaitait. De plus, cela permettait de mieux comprendre ce que l’on expliquait. Bien évidemment, on communiquait aussi lorsque l’on se voyait en vrai, notamment lors des séances de TD ou lorsque nous avions des trous dans l’emploi du temps.

Nous avions également un Drive, puis un Github pour pouvoir avoir accès au code et le mettre à jour avec les modifications faites par chacun. Nous essayions de commenter au fur et à mesure pour que l'on comprenne ce qu'a fait l'autre, et si cela ne suffisait pas, alors on demandait plus d'explications. Lorsque notre emploi du temps nous le permettait (souvent), on travaillait ensemble sur un même ordinateur pour réfléchir à deux cerveaux (et aussi pour repérer plus rapidement des erreurs d'indentation ou de syntaxe et donc d'optimiser du temps). Sinon, en dehors de l'école, nous avancions chacun de notre côté selon les tâches demandées mais nous mettions régulièrement en commun afin que chacun puisse voir où l'autre en était. Ce qui fait que nous comprenions tous les deux le code.

Nous avons également revu tous les commentaires à la fin afin de vérifier que tout était assez clair et esthétique lorsque nous allions rendre le projet.

Concernant l'organisation et les délais prévus pour le projet, nous avons été parfaitement dans les temps et même en avance, ce qui nous a rassuré et ainsi permis de réaliser le bonus (et de s'appliquer sur le magnifique rapport ci présent 😎).

Documentation et outils pour mettre en oeuvre notre projet :

- OpenClassrooms
- Nos cours
- GDB : très utile à la réalisation du projet car lorsque nous étions vraiment bloqués et que nous ne savions pas comment débbugger le programme, GDB nous montrait les endroits où il y avait des problèmes (notamment pour les fautes de segmentation et le fameux "core dumped").
- Youtube
- <https://imagemagick.org/index.php>

Explication des fonctions et captures d'écran :

Commande d'aide -h :

```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh -h
*****
*AIDE*
*****
Veuillez entrer les bornes de vol sous forme de deux entiers compris entre 1 et 4294967295.
Merci de les séparer par un espace.
N'entrez rien d'autre que des chiffres. Pas de point, pas de virugle ni d'autre caractère.
Si besoin de plus d'informations (sur les codes d'erreur par exemple), veuillez lire le readme.
```

Messages d'erreur lors de l'entrée des valeurs :

Code :

```
8 test(){
9     if [ "$1" == "-h" ]; then
10         aide
11         exit 0
12     fi
13     if [ "$#" -ne 2 ] && ! [ "$1" == "-h" ]; then
14         echo "Le nombre d'arguments est invalide"
15         aide
16         exit 1
17     fi
18     if ! [ "$1" == "-h" ] && ! [[ "$1" =~ ^-[0-9]+$ ]]; then
19         echo "Le premier argument n est pas un entier"
20         aide
21         exit 2
22     fi
23     if ! [ "$1" == "-h" ] && ! [[ "$2" =~ ^-[0-9]+$ ]]; then
24         echo "Le second argument n est pas un entier"
25         aide
26         exit 3
27     fi
28     if ! [ "$1" == "-h" ] && [ "$1" -gt "$2" ]; then
29         echo "Le premier argument est superieur au second"
30         aide
31         exit 4
32     fi
33 }
```

Affichage console :

-> Lorsque le nombre d'arguments est invalide

```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh 124
Le nombre d'arguments est invalide
*****
*AIDE*
*****
Veuillez entrer les bornes de vol sous forme de deux entiers compris entre 1 et 4294967295.
Merci de les séparer par un espace.
N'entrez rien d'autre que des chiffres. Pas de point, pas de virugle ni d'autre caractère.
Si besoin de plus d'informations (sur les codes d'erreur par exemple), veuillez lire le readme.
```

```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh 124 1456 15678
Le nombre d'arguments est invalide
*****
*AIDE*
*****
```

-> Lorsque l'on entre un intervalle incorrect

```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh 12 4
Le premier argument est superieur au second
```

-> Lorsque l'on rentre autre chose que des entiers ou que l'on rajoute des caractères

```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh 56b ab45
Le premier argument n est pas un entier
```

```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh 56 ab45
Le second argument n est pas un entier
```

Code :

```
57 if [ "$error" -ne 0 ] ; then    #Si une erreur est detectée dans le programme en c on renvoie le code d'erreur
58     echo "Probleme dans l'execution du programme en c, code d'erreur = "$error""
59     aide
60     exit 5
61 fi
```

Affichage console :

```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh 0 45
Creation des fichiers sources ...
U0 n'est pas un entier superieur a 0
Probleme dans l'execution du programme en c, code d'erreur = 2
*****
*AIDE*
*****
```

Affichage des différentes étapes pour la création des graphiques :

Code :

```
echo "Creation des fichiers sources ..."
```



```
echo "Fichiers sources bien crees"
echo "Creation des graphiques ..."
```



```
echo "Graphiques bien crees les voici :"
```

Affichage console :

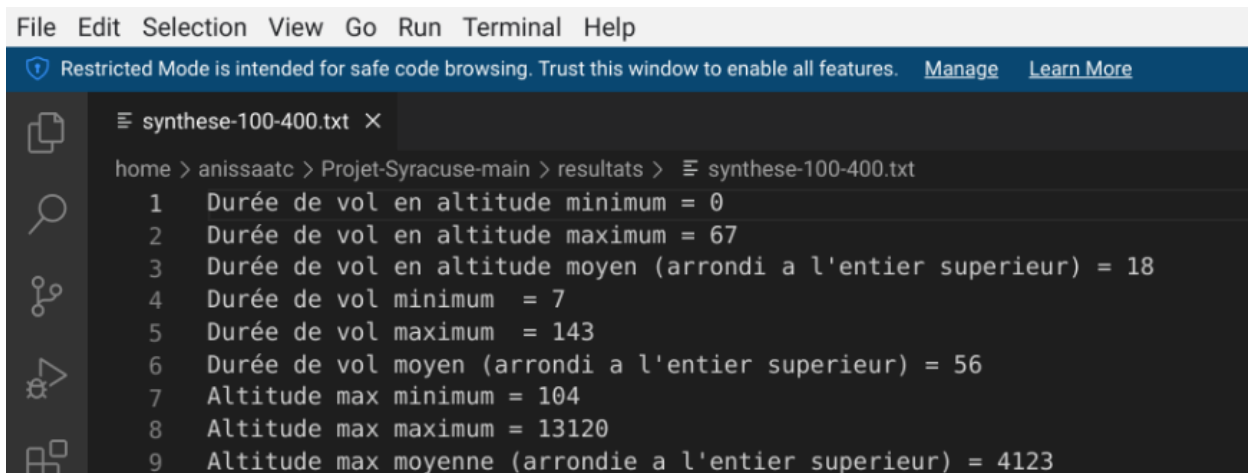
```
anissaatc@penguin:~/Projet-Syracuse-main$ ./Projetbash.sh 100 400
Creation des fichiers sources ...
Fichiers sources bien crees
Creation des graphiques ...
Graphiques bien crees les voici :
```

Création de la synthèse (pour les bornes 100 à 400) :

Code :

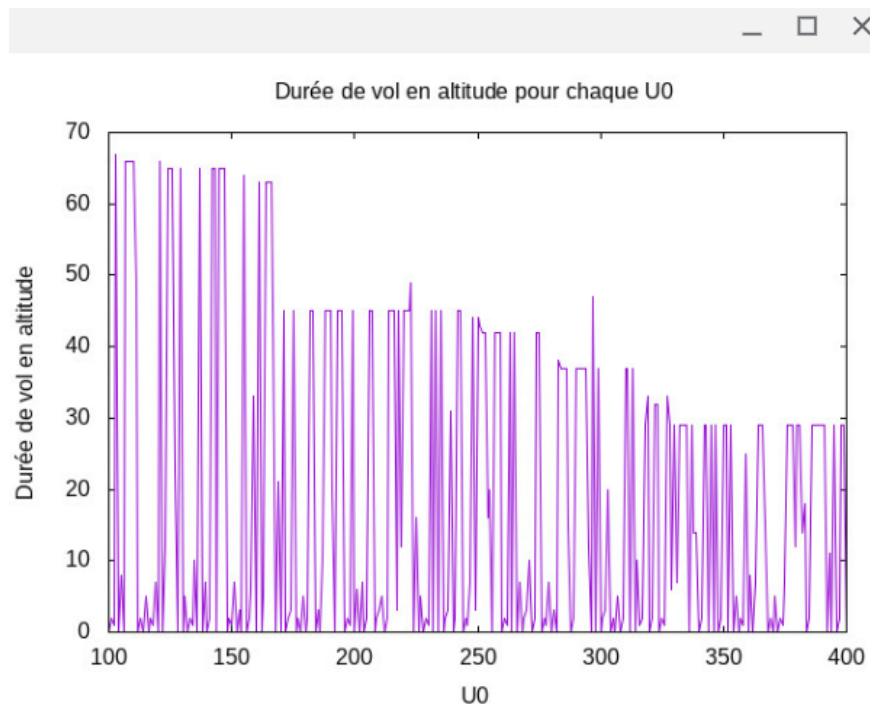
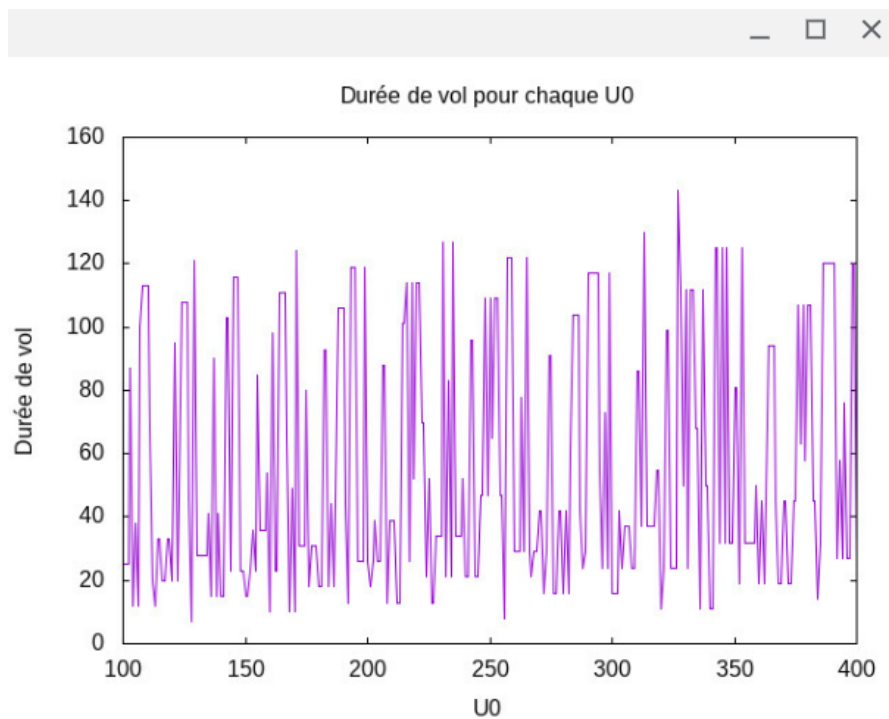
```
i=$((y-x)) #On écrit toutes les informations dans la synthese
echo "Durée de vol en altitude minimum = $min1" >> synthese-$x-$y.txt
echo "Durée de vol en altitude maximum = $max1" >> synthese-$x-$y.txt
echo "Durée de vol en altitude moyen (arrondi a l'entier superieur) = $((($moy1/$i)))" >> synthese-$x-$y.txt
echo "Durée de vol minimum = $min2" >> synthese-$x-$y.txt
echo "Durée de vol maximum = $max2" >> synthese-$x-$y.txt
echo "Durée de vol moyen (arrondi a l'entier superieur) = $((($moy2/$i)))" >> synthese-$x-$y.txt
echo "Altitude max minimum = $min3" >> synthese-$x-$y.txt
echo "Altitude max maximum = $max3" >> synthese-$x-$y.txt
echo "Altitude max moyenne (arrondie a l'entier superieur) = $((($moy3/$i)))" >> synthese-$x-$y.txt
cp synthese-$x-$y.txt ../synthese-$x-$y.txt #On écrase le fichier s'il existe deja (d'où le cp)
```

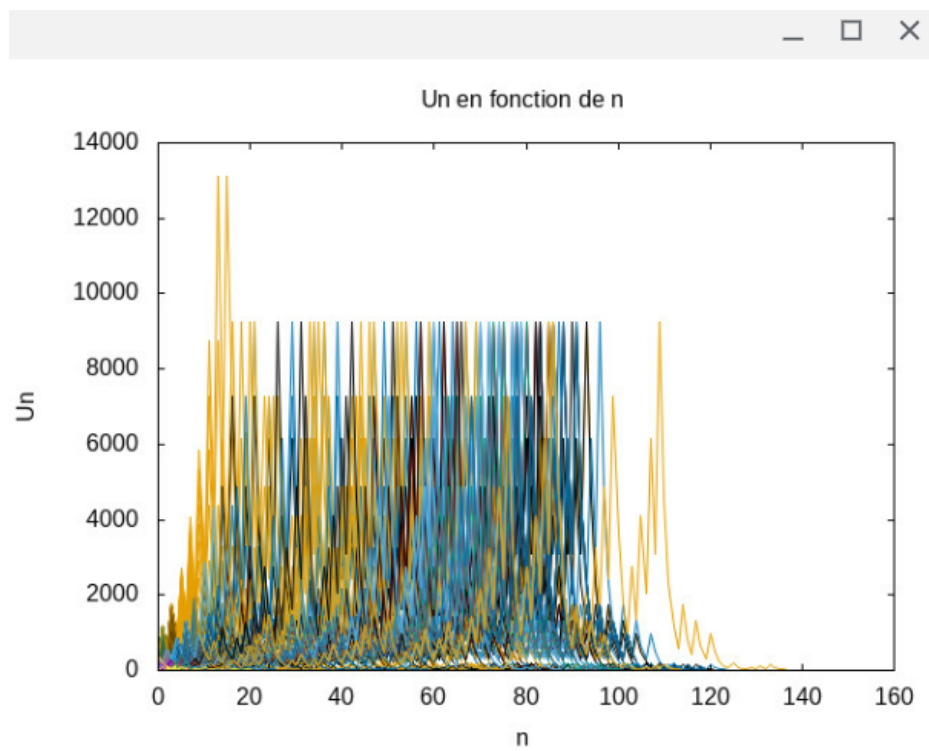
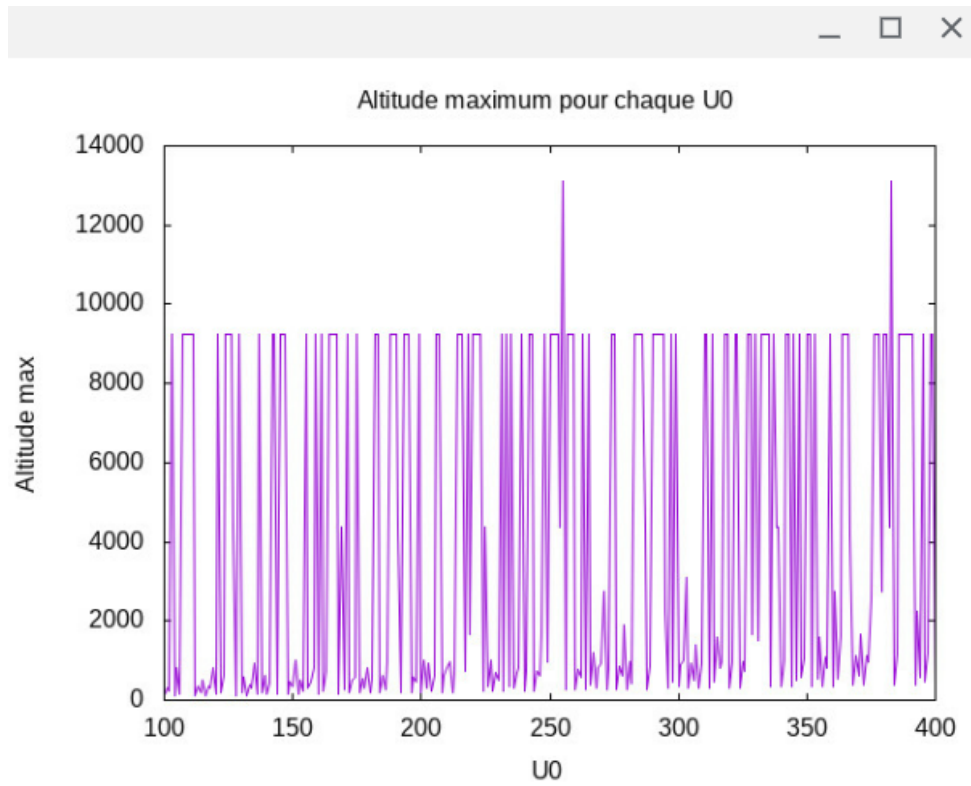
Affichage :



```
File Edit Selection View Go Run Terminal Help
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
≡ synthese-100-400.txt X
home > anissaatc > Projet-Syracuse-main > resultats > ≡ synthese-100-400.txt
1  Durée de vol en altitude minimum = 0
2  Durée de vol en altitude maximum = 67
3  Durée de vol en altitude moyen (arrondi a l'entier superieur) = 18
4  Durée de vol minimum = 7
5  Durée de vol maximum = 143
6  Durée de vol moyen (arrondi a l'entier superieur) = 56
7  Altitude max minimum = 104
8  Altitude max maximum = 13120
9  Altitude max moyenne (arrondie a l'entier superieur) = 4123
```


Graphiques obtenus (toujours pour les bornes 100 à 400) :





Sous répertoire “resultats” créé dans le répertoire du projet :

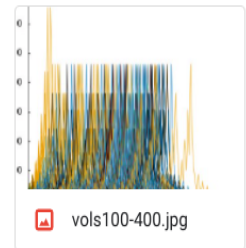
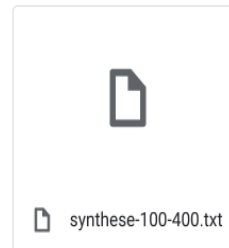
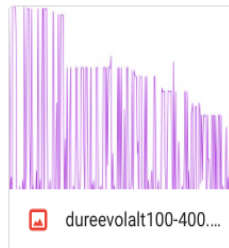
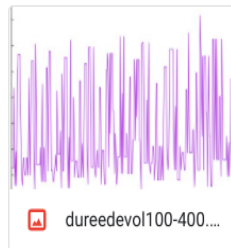
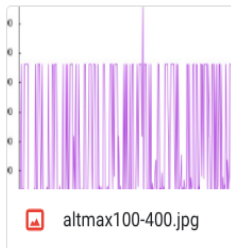
Dossiers

 **resultats**

-> Il contient les 4 graphiques obtenus en format .jpg ainsi que la synthèse en format .txt

Mes fichiers > Fichiers Linux > Projet-Syracuse-main > resultats

Fichiers



Problèmes rencontrés

Nous avons, bien sûr, eu des bugs et des problèmes que nous avons réussi à fixer.

Voici une liste des problèmes rencontrés dans le fichier .C puis dans le fichier bash (.sh) :

- (.C) Choix du type de variable pour les entiers saisis : unsigned long choisi (après débat dans l'équipe) pour un meilleur résultat
- (.C) Convertir les caractères en chiffres : réglé en utilisant strtoul
- (.C) Comprendre comment réaliser la fonction récursive (puis la coder)
- (.sh) Nous n'avons pas utilisé mktemp que l'on connaissait, nous avons donc "manuellement" créé un répertoire temporaire qui se supprime après
- (.sh) Syntaxe de gnuplot assez difficile
- (.sh) Problèmes au niveau des tests car un membre dispose d'un chromebook où linux bug parfois (notamment pour la suppression du tmp, il a fallu installer gnome)
- (.sh) Quelques difficultés avec display

Conclusion

En somme, ce projet a été l'occasion pour nous, encore une fois, de travailler en équipe. Dans l'ensemble, nous avons travaillé de manière efficace, sans conflit ou autre. Chacun a respecté ses tâches et nous avons été très communicants, ce qui a été une clé pour la réussite de ce projet.

Nous avons, encore une fois, pu développer nos connaissances et savoirs-faire en C (surtout pour la récursivité où on a pris un bon bout de temps) et acquérir des connaissances en script shell (commandes linux, pipes, découverte de gnuplot...). Nous sommes donc satisfaits de ce projet.