Rapport / README Projet Syracuse

Présenté par Adrien MACHNIK et Gautier LESZNEWSKI Pré-ING 2 MI Groupe 2

Le projet Syracuse consistait à exécuter la suite de Syracuse en utilisant un code en langage C et une script shell.

Le langage C doit prendre en paramètre un premier terme entier (qu'on appellera u0) ainsi qu'un nom de fichier dans lequel les données calculées apparaîtront.

Le script en Bash doit quant à lui prendre en paramètres les valeurs minimales et maximales de u0. Il doit par la suite exécuter le programme C pour chaque valeur entre les minimums et maximums rentrés en paramètres. Le script doit ensuite créer plusieurs courbes : courbes de données, courbe de l'altitude maximale, courbe de la durée de vol et courbe de la durée de vol en altitude (toutes ces courbes en fonction des différents Un entre le minimum et le maximum).

- **Explication partie C:**

➔ Fonction main :

- La fonction main vérifie qu'il y a le bon nombre d'arguments en paramètres (appel, u0, nom du fichier) il va ensuite ouvrir un fichier avec le mode lecture/écriture.

```
if(argc !=3) exit(EXIT_FAILURE); // Si
int Un = atoi(argv[1]);

FILE* fichier = fopen(argv[2], "w+");
if (fichier == NULL) printf("error");
```

- Après avoir effectué ce travail, il va appeler la fonction syracuse et lui donne en paramètres le nombre de départ et le fichier dans lequel on va écrire les valeurs.

```
syracuse(Un, fichier);
```

➔ Fonction syracuse :

- La fonction va dans un premier temps initialiser l'altitude maximale, le temps de vol, un compteur pour compter le nombre d'étapes en altitude, un pour le premier terme et un pour stopper un compteur spécial

```
int altitudeMax = 0; //
int i = 0; // Temps de
//int Un = 0; //Initial
int UCompteur = 0; // P
int IncrUCompteur = 1;
int UPremier = 0; //pre
```

- La fonction marquera ensuite dans le fichier en paramètre chaque altitude en fonction de l'étape

```
fprintf(f, "n Un\n"); // marc
altitudeMax = Un;
UPremier = Un;
fprintf(f,"%d %d\n", i, Un);
i++;
```

- Par la suite, la fonction syracuse va effectuer la fameuse suite de syracuse (disjonction des cas en fonction de Un pair ou impair)

```
while (Un > 1) {

  if (Un % 2 == 0) { // Cas Un pair

    Un = (Un / 2);
    fprintf(f,"%d %d\n", i, Un);
    i++;
  }
  else { // Cas Un impair
    Un = ((Un * 3) + 1);
    fprintf(f,"%d %d\n", i, Un);
    i++;
  }
```

- Elle calcule ensuite l'altitude maximale

```
if (altitudeMax < Un) {
    altitudeMax = Un; //
}
```

- Puis le calcul du temps de vol au dessus de l'altitude de départ

```
if (Un < UPremier) {
  IncrUCompteur = 0;
}
if (Un >= UPremier) {
  UCompteur += IncrUCompteur;
}
```

- Et pour finir, la fonction affiche l'altitude maximale, la durée de vol et la durée d'altitude :

```
fprintf(f,"Altitude max:%d\n", altitudeMax); // a
fprintf(f,"Duree de vol:%d\n", i); // affiche la
fprintf(f,"Durée de l'altitude:%d\n", UCompteur);
```

- **Explication partie Bash:**

Le script est composé de deux Grandes parties.
La première, les fonctions et la seconde, le main.

➔ Les fonctions :

```
function recup_Un {
    head -n-3 $1 | tail -n+2 >> $2 && echo >> $2
}
function recup_altitudemax {
    echo "$3 $(tail -n3 $1 | head -n1 | cut -d':' -f2)" >> $2
}
function recup_dureevol {
    echo "$3 $(tail -n2 $1 | head -n1 | cut -d':' -f2)" >> $2
}
function recup_dureealtitude {
    echo "$3 $(tail -n1 $1 | cut -d':' -f2)" >> $2
}
function max_min_altitudemax {
    echo "Altitude max de toutes les altitudes max de tous les U0: $(sort -k2n $1 | cut -d' ' -f2 | tail -1)" >> $2
    echo "Altitude min de toutes les altitudes max de tous les U0: $(sort -k2n $1 | cut -d' ' -f2 | head -1)" >> $2
}
function max_min_dureevol {
    echo "Durée de vol max de tous les U0: $(sort -k2n $1 | cut -d' ' -f2 | tail -1)" >> $2
    echo "Durée de vol min de tous les U0: $(sort -k2n $1 | cut -d' ' -f2 | head -1)" >> $2
}
function max_min_dureealtitude {
    echo "Durée d'altitude max de tous les U0: $(sort -k2n $1 | cut -d' ' -f2 | tail -1)" >> $2
    echo "Durée d'altitude min de tous les U0: $(sort -k2n $1 | cut -d' ' -f2 | head -1)" >> $2
}
```

Les fonction **“recup_*”** récupère les données de tous les U0

La fonction *recup_altitudemax* va envoyer sur la même ligne notre **U0 =”$1**” et notre altitude maximum :

`tail -n3 $1` permet de prendre les trois dernières lignes de notre fichier **“$1”** en question

`| head -n1 |` permet de prendre la première ligne de notre fichier

`cut -d':' -f2` permet de prendre seulement notre donnée de droite

`>> $2` permet de renvoyer tous les résultats précédent dans notre fichier destinataire **“$2”**

Le processus est le même pour les autres fonctions de type **“récup_*”**

Nos fonction de tries quant à elles s'appuient sur la commande **“sort”:**

```
$(sort -k2n $1 | cut -d' ' -f2 | tail -1)" >> $2
```

```
$(sort -k2n $1 | cut -d' ' -f2 | head -1)" >> $2
```

On va tout d’abord trier la colonne 2 en fonction des nombres, puis prendre seulement colonne 2, donc notre partie droite et enfin récupérer la donnée qui se situe en première ligne, ou en dernière ligne pour avoir respectivement notre valeur min et max.

- ➔ fonction main :
  - ◆ Vérification des arguments, qui doit être de trois.

```
if [ "$#" -ne 2 ]
then
    echo "Le nombre d'argument doit être de 3"
    exit
fi
```

- Comparaison d'ancienneté entre l'exécutable et le .c. Si l'exécutable est plus vieux, il va le recompiler, la comparaison se fait grâce à **"-nt".**

```
if [ "$SRC" -nt "$EXE" ]
then
    echo "Recompilation de l'executable $EXE"
    $CC -o $EXE $SRC
fi
```

- Création du fichier temporaire "Data"

```
#Creation d un fichier temporaire
mkdir -p Data
```

- On va ensuite effectuer une boucle qui va s'effectuer en fonction de nos bornes dans laquelle il y aura l'exécution du .c ainsi que l'appel de nos fonctions **"recup_*"** et pour finir la suppression des fichiers en fonction de nos U0.

```
for (( i=$1; i<=$2; i++))
do
    #execute le programme C ./exe U0 fileU0.dat
    ./$EXE $i Data/file$i.dat

    # on recupere les données dans des fichiers temporaires à l'aide de nos fonctions

    #récupération de toutes les suites en fontion de n
    recup_Un Data/file$i.dat Data/toutes_suites.dat

    #récupération de toutes les altitudes max
    recup_altitudemax Data/file$i.dat Data/toutes_altitude_max.dat $i

    #récupération de toutes les durées de vols
    recup_dureevol Data/file$i.dat Data/toutes_duree_vol.dat $i

    #récupération de toutes les durées d altitudes
    recup_dureealtitude Data/file$i.dat Data/toutes_duree_altitude.dat $i

    #On supprime les fichiers dès lors qu'ils sont créés
    rm Data/file$i.dat
done
```

- Création du fichier "Graphes"

```
mkdir -p Graphes
```

- Lancement de gnuplot ainsi que de ses commandes qui nous permettent d'effectuer nos graphes

```
gnuplot -persist <<-EOFMarker
    set terminal jpeg
    set output "Graphes/Graphe_toutes_suites[$1;$2].jpeg"
    set title "Un en fonction de n pour U0 dans [$1;$2]"
    set xlabel "n"
    set ylabel "Un"
    plot "Data/toutes_suites.dat" w l title "vols"
    reset
    set terminal jpeg
    set output "Graphes/Graphe_duree_vol[$1;$2].jpeg"
    set title "Durée de vol en fonction du U0 dans [$1;$2]"
    set xlabel "U0"
    set ylabel "Nombre d'occurences"
    plot "Data/toutes_duree_vol.dat" w l title "durée vol"
    reset
    set terminal jpeg
    set output "Graphes/Graphe_altitude_max[$1;$2].jpeg"
    set title "Altitute maximum atteinte en fonction de U0 dans [$1;$2]"
    set xlabel "U0"
    set ylabel "Altitude Maximum"
    plot "Data/toutes_altitude_max.dat" w l title "altitude"
    reset
    set terminal jpeg
    set output "Graphes/Graphe_duree_altitude[$1;$2].jpeg"
    set title "Durée de vol en altitude en fonction de U0 dans [$1;$2]"
    set xlabel "U0"
    set ylabel "Nombre d'occurence"
    plot "Data/toutes_duree_altitude.dat" w l title "durée altitude"
    reset
EOFMarker
```

- Et pour finir l'appel de nos fonctions de trie:

```
#--------------------------BONUS--------------------------#
# bonus faire la synthese min max moyenne

max_min_altitudemax Data/toutes_altitude_max.dat synthese-min-max.txt

max_min_dureevol Data/toutes_duree_vol.dat synthese-min-max.txt

max_min_dureealtitude Data/toutes_duree_altitude.dat synthese-min-max.txt
```

READ ME

➔ Il va falloir rendre notre .sh exécutable en effectuant ceci :

```
chmod u+x syracus.sh
```

➔ Pour l'executer il va ensuite falloir tapper:

```
./syracus.sh $1 $2
```

tel que $1 correspond à notre borne inf et $2 notre borne sup [$1;$2].

➢ *Exemple* :

```
./syracus.sh 100 500
```

Le script va créer un dossier temporaire nommé **"Data"** dans lequel il va stocker tous les fichiers temporaires utilisés pour faire les graphiques.

❖ *Avant*

```
syracus  syracus.c  syracus.sh
```

❖ *Après*

```
Data  syracus  syracus.c  syracus.sh
```

Ce dossier ne sera pas visible car supprimé à la fin du script, tout comme les fichiers d'ailleurs.
Un dossier **"Graphes"** va se créer dans lequel nous stockerons ces derniers ainsi qu'un fichier .txt **"synthese-min-max.txt"** ou seront mis les valeurs maximum/minimum de chaque paramètre altitude maximum, durée de vol et durée de vol en altitude.

```
Graphes  synthese-min-max.txt  syracus  syracus.c  syracus.sh
```

➢ *Exemple de Graphes pour des valeurs comprises entre [100;500]*