Rapport TP3 Perfectionnement C

DE JESUS MILITAR Dylan

7 mars 2021

1 Introduction

Les objectifs de ce TP étaient l'approfondissement de la manipulation graphique à l'aide la librairie ncurses, par le biais de l'implémentation en langage C d'un chronomètre ayant plusieurs fonctionnalités. Ce projet visait également à se familiariser avec la mise en place de pré assertions et de gestion d'erreurs en tant que bonne habitude et comportement de programmation.

2 Exercice 1

2.1 PremPart.c

Le programme PremPart.c regroupes les questions 1, 2. Dans ce programme j'ai simplement pris les valeurs en secondes et microsecondes de debut et de fin, les ait convertis en milisecondes puis soustrait entre elles (entre debut et fin) puis j'ai additionné ces résultats pour obtenir le temps en ms afin d'obtenir la fonction intervalle-ms Pour les fonctions de conversions il a simplement suffit d'implémenter les valeurs de conversions entre unités et d'appliquer l'opération modulo selon le nombre maximal qu'il devaient atteindre(ex : modulo 60 pour les secondes et minutes, et modulo 100 pour les centièmes et heures)

2.2 ChronoSimple.c

Le programme ChronoSimple.c concerne la question 3. J'ai créé une boucle, dans laquelle j'ai constamment toutes les 500 ms (usleep(50000)) reprise des nouvelles valeurs de temps. J'ai ensuite à chaque fois grâce à une fonction affichée de manière formatée à l'aide des fonction de conversions de la question 2.

2.3 ChronoMoyen.c

Le programme ChronoMoyen.c concerne la question 4. Dans cette partie il a suffit d'implémenter l'algorithme donné à l'aide des outils de PremPart.c. De plus nodelay(stdscr, TRUE) ajoué un rôle très important

3 Exercice 2

L'exercice 2 est représenté par un unique programme : Ex2.c
.La fonction d'initialisation est simplement un renoie d'un objet de struture Chronometre pour lequel

ses champs ont été initialisés à 0 sauf "avertissement" initialisé à 25 secondes .Par oubli cette fonction porte un autre nom dans le programme, définie dans ChronoMoyen.c : formatAffichageGraphique() ; Elle s'appuie sur le fonctionnement de la fonction d'affichage en sortie standard de ChronoSimple.c Les autres fonctions sont des fonctions graphiques tout les points sont précisés dans le programme

4 Exercice 3

4.1

Cette exercice donne lieu au programme Chrono.c

1. Il a simplement suffit de mettre une condition, si la touche pressée est égale à 'q' alors on la valeur de la variable (ici etat-programme) dans la condition du while doit être fausse pour sortir du programme

2. En utilisant une condition, si la touche pressée vaut 'r' alors on met les champs etat, nb-tour, indice-der-tour et duree de la structure du chronometre à 0

3. Sur appui de la touche 't' par l'utilisateur, le programme appelle la fonction ajoute-tour() afin de bien enregistré dans le champ temps-tour du chronometre la valeur de duree en première place du tableau

4. Si la valeur de duree est plus grande et que celle d'avertissement, et que c'est la première fois que c'est le cas, alors on fait appelle à l'avertisseur visuel qu'est affiche-flash() pour prévenir l'utilisateur

5. Pour installer les touches F1, F2,. . . ..F6 , on met keypad(stdscr, TRUE) afin de débloquer les macros KEY-F(i) avec i étant un entier supérieur ou égal à 1 permettant de choisir quelle touche est appuyée par l'utilisateur. Selon la touche appuyée on appelle la fonction conversionMilisecondes() avec en argument le type de temps (ex : 'H' si F1 ou F2 pour les heures avec F1 qui incrément et F2 qui décrémente).

5 Compilation et utilisation des programmes

Tout d'abord chaque programme possède un main permettant d'être executé et testés mis à part du projet dans sa globalité, pour cela il suffit simplement de retirer if 0. . . endif autour du main(), et de compiler normalement à l'aide de : gcc Prog.c -ansi -pedantic -Wall -o Prog (-lncurses si besoin)

Pour pouvoir compilé et exécuté le projet dans sa globalité, il faut se placer au niveau de Chrono.c qui est le programme permettant de lier toutes les sous-parties du projet. On compile alors à l'aide de la commande : gcc Chrono.c Ex2.c PremPart.c ChronoMoyen.c -ansi -pedantic -Wall -o Chrono -lncurses. Puis d'executer avec : ./Chrono
Sino n pour chaque programme il faut utiliser les commandes de compilation suivantes :

PremPart.c -> gcc PremPart.c -ansi -pedantic -Wall -o PremPart (en enlevant le if 0..endif autour du main)
ChronoSimple.c -> gcc PremPart.c ChronoSimple.c -ansi -pedantic -Wall -o ChronoSimple (en enlevant le if 0..endif autour du main de ChronoSimple)

ChronoMoyen.c -> gcc PremPart.c ChronoMoyen.c -ansi -pedantic -Wall -o ChronoMoyen -lncurses (en enlevant le if 0..endif autour du main de ChronoMoyen)

Ex2.c -> gcc PremPart.c ChronoMoyen.c Ex2.c -ansi -pedantic -Wall -o Ex2 -lncurses (en enlevant le if 0..endif autour du main de Ex2)

6 Difficultés rencontrées

Lors de ce projet, j'ai rencontré une grande difficulté à pouvoir le terminer en temps et en heures. En effet, alors que je travaillais sur ma partie, mon binôme n'a pas travaillé sur la sienne ceci me bloquant pour le passage à l'exercice 3. De plus en toute fin de semaine j'ai été contraint de devoir travailler complètement seul sur ce projet, et de le faire de A à Z, car mon partenaire n'a plus donné de nouvelles depuis. Le manque de temps à ainsi été extrêmement difficile à gérer ainsi que la difficulté de devoir être seul sur un projet entier. Cela a également joué sur l'organisation de mon temps de travail, et de ce projet.

7 Conclusion

Ce projet m'a permis de mener à bien la corrélation entre des élément graphiques et de gestion du temps. La conception d'un chronomètre m'a ainsi permis d'apprendre une nouvelle habitude de programmation vu en cours qu'elle la programmation modulaire.