Institut Paul Lambin 20 mars 2019

I2180 : Interrogation de Programmation Système

J. Vander Meulen, B Frank, O Choquet, A Legrand

Informatique de Gestion - Bloc 2

Durée de l’interrogation : 90 minutes

**Consignes importantes** :

* **Redémarrez le PC et choisissez Ubuntu**
* **Utilisez la feuille de login pour vous connecter**
* **Notez sur cette page de garde vos nom et prénom et votre login Linux.**
* **Pour connecter le drive Z:\ (cf. capture d’écran en page 2)**
  + Ouvrir un explorateur de fichiers et cliquer sur « Connexion à un serveur »

(tout en bas à gauche)

* + Entrez l’adresse samba suivante :

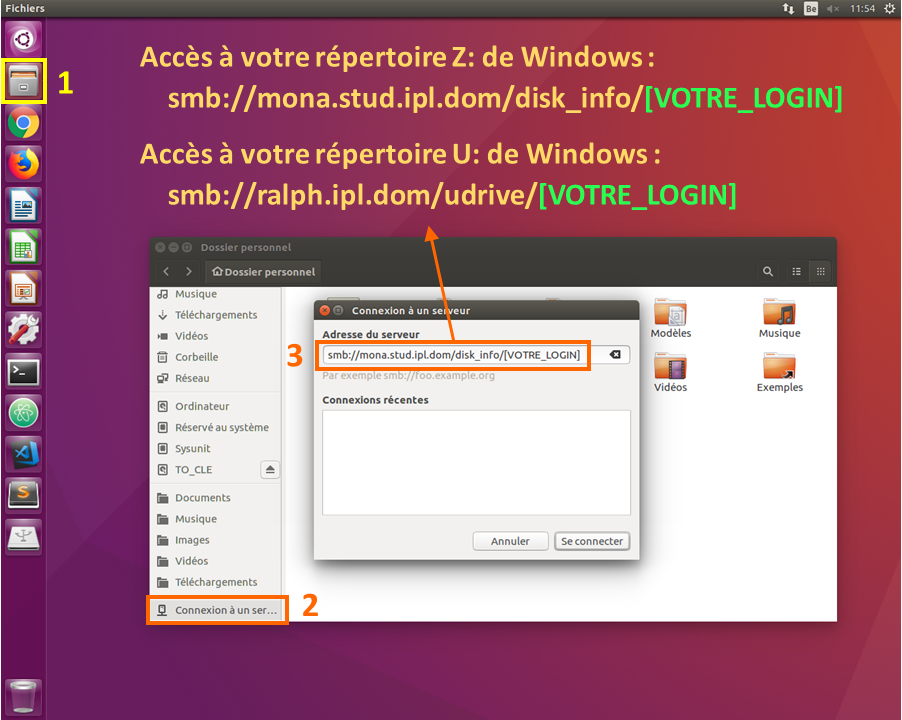
**smb://mona.stud.ipl.dom/disk\_info/[*LOGIN FOURNI*]**

* + Votre répertoire s’ouvrira dans une nouvelle fenêtre.
* **Créez à la racine du drive Z : un répertoire dont le nom suit la nomenclature suivante : NOM\_PRENOM.**

**Par exemple : SERONT\_GREGORY**

* **A la fin de l’interrogation, placez tous vos fichiers dans ce répertoire.**
* **Nous vous demandons de faire particulièrement attention à ce que votre code ne produise par d’erreurs de compilation. Notez que nous vous demandons de compilez votre code avec les flags utilisés lors du cours :**

cc –D\_DEFAULT\_SOURCE -std=c11 -pedantic   
 -Wall -Wvla –Werror



Cette page est volontairement vide.

# Question 1

Le code à compléter de trouve dans le fichier « Question\_1.c ».

Le fichier nommé « data.bin » contient les données à traiter. C’est un fichier binaire qui contient d’abord un int indiquant le nombre d’enregistrements qui y sont stockés. Il contient ensuite chacun des enregistrements. Chaque enregistrement se présente suivant la structure suivante :

struct record {char nom[64] ; char prenom[64] ; char sexe}  
où sexe prend l’une des deux valeur ‘H’ ou ‘F’.

**Vous devez générer le fichier « data.bin »** en exécutant les deux commandes suivantes dans votre console :

cc -std=c11 -pedantic -Wall -Wvla –Werror Remplissage.c

./a.out

Le fichier « data.bin » sera redirigé sur l’entrée standard lors de l’instruction d’exécution du programme sur la ligne de commande (p.ex. : ./a.out < data.bin).

Le programme lit tous les enregistrements et les stocke en mémoire en mettant les champs « nom » en majuscules. Le père crée un processus fils auquel il délègue le traitement des personnes de sexe féminin, tandis qu’il traite lui-même les personnes du sexe masculin.

Lorsque la fin du fichier est atteinte, le fils écrit le nombre d’enregistrements qu’il a traité dans le fichier binaire « resultat.bin ». Puis il écrit chacun de ces enregistrements dans ce fichier. Lorsque le fils a terminé, le père écrit le nombre d’enregistrements qu’il a traité dans le fichier binaire « resultat.bin ». Ensuite, le père écrit chacun de ces enregistrements dans ce fichier.

Finalement, le père lit tous les enregistrements présents dans le fichier « resultat.bin » et les écrit sur la sortie standard.

**Ce qui vous est demandé (marqué /\* A COMPLETER \*/ dans « Question\_1.c ») :**

La communication entre le père et le fils se fera via un pipe. Vous devez compléter entièrement le code concernant la création, la gestion et l’utilisation de ce pipe.

On vous demande aussi de compléter toutes les instructions de lecture et d’écriture, en utilisant exclusivement les appels systèmes read(…) et write(…), à l’exception des dernières opérations du père qui vous sont fournies.

On vous demande enfin de veiller à ce que le père attende la fin de son fils avant de commencer à écrire dans le fichier « resultat.bin ».

Pour vérification, votre programme doit obtenir l’output suivant lorsque le fichier « data.bin » est redirigé vers l’entrée standard :

Moi, le père, j’ai traite 6 enregistrements de sexe = ‘H’

Moi, le fils, j’ai traite 4 enregistrements de sexe = ‘F’

Nom : DURE Prenom : Laure Sexe : F

Nom : JAVEL Prenom : Aude Sexe : F

Nom : DIOTTE Prenom : Kelly Sexe : F

Nom : POTENPOCHE Prenom : Jessica Sexe : F

Nom : ROGNE Prenom : Yves Sexe : H

Nom : COEUR Prenom : Marc Sexe : H

Nom : FAITMAL Prenom : Oussama Sexe : H

Nom : TARE Prenom : Guy Sexe : H

Nom : DEUF Prenom : John Sexe : H

Nom : ASSIN Prenom : Marc Sexe : H

En plus du fichier « Question\_1.c », nous vous demandons de compléter le fichier « Makefile » afin de produire l’exécutable « Question\_1»

# Question 2

Dans cette question, vous allez construire un programme C qui simule deux composants imaginaires d’un train. Un train exécute deux programmes, un capteur qui envoie des signaux à un serveur et un serveur qui traite les signaux reçus.

Un train démarre et s'arrête toujours dans une gare. Lorsqu'il quitte une gare, il reçoit un signal "SIGUSR1". Lorsqu'il entre en gare, il reçoit un signal "SIGUSR2". Lorsqu’il termine son trajet, il reçoit un signal "SIGQUIT".

Un trajet est valide s’il génère une suite de signaux de la forme < SIGUSR1, SIGUSR2, SIGUSR1, SIGUSR2, …, SIGQUIT >. C’est-à-dire une suite qui alterne les signaux "SIGUSR1" suivi de "SIGUSR2" et qui termine par "SIGQUIT". Si le trajet génère une suite de signaux qui n’est pas de cette forme, il est invalide. Par exemple :

1. le trajet représenté par la suite <SIGUSR1, SIGUSR2, SIGUSR1, SIGUSR2, SIGQUIT> est valide ;
2. le trajet représenté par la suite <SIGUSR1, SIGUSR2, SIGUSR1,  SIGQUIT> est invalide;
3. le trajet représenté par la suite <SIGUSR1, SIGUSR1, SIGUSR1, SIGUSR2, SIGQUIT> est invalide ;
4. le trajet représenté <SIGQUIT> est valide.

Nous vous demandons d’écrire une partie du simulateur de train. Pour ce faire, complétez le ficher "train.c". Votre programme doit créer un fils. Le père jouera le rôle du capteur et le fils le rôle du serveur.

1. Le père doit lire sur "stdin" une suite de lignes. Il n’y a que deux lignes possibles :
   1. "q\n" qui signifie que le train quitte une gare
   2. "e\n" qui signifie que le train entre dans une gare

Vous pouvez supposer que ces lignes respectent toujours ce format. Vous ne devez pas faire de vérification sur format des lignes. Après chaque ligne lue, le père doit envoyer un signal à son fils puis s’endormir 1 seconde. Lorsque le père lit une ligne vide (“CTRL + D”), il envoie à son fils un signal "SIGQUIT".

1. Le fils reçoit des signaux de type "SIGUSR1", "SIGUSR2", ou "SIGQUIT".  Lorsqu’il reçoit un signal de type "SIGQUIT", il affiche "yes" sur « stdout » si le trajet est valide ; il affiche "no" sinon. Ensuite, il se termine.

En plus du fichier "train.c", nous vous demandons de compléter le fichier "Makefile" afin de produire l’exécutable "train".

Les fichiers texte "ex1", "ex2" et "ex3" reprennent les trois premiers exemples ci-dessus. Vous pouvez les utiliser en exécutant le programme de la manière suivante : "./train < ex1"

# Brouillon :