Examen « systèmes centralisés I » 2020-2021

Question 1 : **1 point**

La commande ls –l liste sous le format suivant :

Droits nombre propriétaire groupe taille mois jour année/heure nom

Dire en une ou deux phrases ce que fait la commande : ls –l nom_fichier | cut –d' ' –f5

Question 2 : 1 point

La commande « cat », exécutée sans argument, lit le flot de données entré au clavier et le réaffiche à l'écran. Elle s'arrête lorsque « CtrlD » est entré (équivalent à « Fin de fichier »). La commande « expr » permet d'évaluer une expression arithmétique : a=`expr $b + $c` enregistre dans a la somme des variables b et c (si elles représentent des valeurs entières).

Dire ce que fait le script en face, et en donner la sortie pour une entrée clavier composée de : aa bb cc
et terminée par CtrlD
	count=0
	for mot in `cat`; do
	  count=`expr $count + 1`
	  echo "$count: $mot"
	done

Question 3 : 2.5 points - Ecrire le script « volume.sh » qui lit un flot de données au clavier, composé de noms de fichiers, de dossiers, …, et calcule le volume total des ***fichiers*** présents dans le flot en entrée et dont la taille est supérieur à la valeur passée en argument.

Question 4 : 1.5 points

A- Donner la ligne de commande qui calcule le volume total des fichiers présents dans un répertoire donnée et dont la taille est supérieure à 1000 octets, et expliquer ce que fait chaque partie de cette ligne de commande.

B- Donner la ligne de commande qui permet de calculer le volume total des fichiers présents dans l'arborescence d'un répertoire donné et dont la taille est supérieure à 1000 octets ; et expliquer ce que fait chaque partie de cette ligne de commande.

Question 5 : 2 points - Soit le code suivant :

```
// Zone 0
int main() {
   // Zone 1
   int retour = fork() ;
   if (retour < 0) {   printf("Erreur fork\n") ; exit(1) ; }
   else if (retour == 0) {    // Zone 2
                sleep(10); exit(0); }
   else {         // Zone 3
                retour = fork() ;
                if (retour < 0) {   printf("Erreur fork\n") ; exit(1) ; }
                else if (retour == 0) {    // Zone 4
                        sleep(30); exit(0); }
                else {   Zone 5
```

```
                    sleep(20); return EXIT_SUCCESS ; }
        }
}
```

A- Combien y aura-t-il de processus à l'exécution de ce code, et quel seront leurs liens de parenté. On les nommera processus1, processus 2, …, selon l'ordre de création.

B- Dans quel état sera chacun de ces processus 15 secondes après le début d'exécution ?

C- Dans quel état sera chacun de ces processus 25 secondes après le début d'exécution ?

<u>Question 6</u> : 2 points

A- Un processus père doit attendre la fin de ses fils. Donner la séquence de code qu'il faut ajouter pour effectuer cette attente dans le cas où le père n'a rien à faire d'autre ; et en indiquer l'endroit ***(Zone 0, Zone1, Zone2, Zone3, Zone4, Zone5).***

B- Donner les séquences de code qui remplacent la réponse A, dans le cas où le père a d'autres traitements à faire et ne souhaite pas rester bloqué sur ce point ; et en indiquer l'endroit. Expliquer le mécanisme mis en jeu dans cette solution.

<u>Question 7</u> : **2 points**

Processus1 souhaite ne pas être dérangé par le signal SIGINT. Processus2 préfère s'endormir pendant 5 secondes lorsqu'il reçoit ce signal. Processus3 s'arrête.

Donner les séquences de codes à ajouter pour les satisfaire (en indiquer l'endroit).

<u>Question 8</u> : **4 points**

On souhaite faire communiquer ces processus de façon circulaire : processus 1 envoie au processus 2 un message de N octets ; ce dernier envoie au processus 3 le message reçu tronqué du dernier octet ; … ; le dernier processus envoie au processus1 le message reçu tronqué du dernier octet. A chaque retransmission, le message est tronqué d'un octet. Le processus qui reçoit le dernier caractère arrêtera la retransmission.

On utilisera un tableau de caractère « buffer » supposé déclaré et initialisé.

A- Donner les séquences de code à ajouter, en en indiquant l'endroit, pour permettre cette communication. On nommera les structures utilisées : procXtoprocY (X et Y = chiffres).

B- De quelle manière les processus vont-ils savoir que leur émetteur a arrêté d'envoyer et comment sortiront-ils de l'attente de messages qui n'arriveront plus ?

<u>Question 9</u> : 4 points

Deux processus ont des contraintes particulières : processus1 effectue d'autres traitements et veut aller vérifier de façon non bloquante, toutes les 5 secondes, l'arrivée éventuelle d'un message. Processus2 souhaite être libre pour faire autre chose et demande d'être notifié en temps réel des messages qui lui sont destinés.

A- Donner une solution pour le processus1 et en fournir les séquences de code (en précisant l'endroit)

B- Faire de même pour le processus2