Contrôle 1 08/10/2014

Nom: _____

Vous disposez d'1h30. Aucun document autorisé. Toute tentative de communication avec un voisin ou l'extérieur peut être sanctionnée. Dispositifs électroniques interdits.

Question:	Cours	Bash	Création de Processus	Ordonnancement
Points:	4	4	6	4

Questions de Cours

1. (2 points) Expliquez la différence entre:

```
ls > liste.txt ; cat liste.txt  
ls > liste.txt & cat liste.txt
```

Expliquez quel résultat on est censé d'obtenir dans le deux cas en sachant que le fichier liste.txt n'existe pas dans le système de fichiers.

Ils doivent savoir que avec ; la deuxième commande est exécutée après la fin de la première. Dans ce cas là, la première suite de commandes fonctionne et montre le résultat de la commande ls redirigée sur liste.txt.

La deuxième suite de commande est censé échouer car liste.txt n'existe pas encore lors que cat est lancé.

1 point s'ils donnent la différence entre ; et &, 1 point s'ils voient que la deuxième échoue

2. (2 points) Donnez la définition et expliquez à quoi sert le BCP.

Vu en cours:

(0,5) BCP = Bloc de Contrôle de Processus

(1) contient les informations nécessaires au SE pour le contrôle et l'exécution du processus

(0,5) en plus s'ils disent aussi qu'il contient de deux parties, table des processus et user_area

Programmation Bash

3. (4 points) Ecrire un programme Bash qui prend un paramètre: si le paramètre est un nom de répertoire, il compte le nombre de sous-répertoires contenus dans le répertoire et affiche le resultat, sinon il affiche un message d'erreur.

0,5 point s'ils font le test -d correctement
0,5 point pour les `` autour du ls \$1
1 point pour structure correcte du if et du for
1 point s'ils enchainent \$1 et \$i
1 point si l'expression arithmetique est correcte

Ma solution:

```
#!/usr/bin/bash
```

```
sum=0
if [ -d $1 ]
then
    for i in `ls $1`
    do
        if [ -d "$1/$i" ]
        then
            sum=$((sum+1))
        fi
    done
    echo "$1 contient $sum sous-repertoires"
else
    echo "$1 n'est pas un repertoire"
fi
```

Création de Processus

4. (6 points) Ecrire un programme C qui crée deux fils: le premier affiche les entiers de 1 à 50, le deuxième les entiers de 51 à 100. Assurez-vous que l'affichage soit dans l'ordre.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

int main(void) {
    pid_t pid;
    int i;
    pid = fork();
    if(pid < 0) /* Echec de fork */ exit(1) ;

    if (pid == 0)
    {
        /* fils1 */
        for (i = 1; i <= 50; i++)
            printf("%d\n", i);
        return 0;
    }

    wait(NULL);

    pid=fork();
    if(pid < 0) /* Echec de fork */ exit(1) ;

    if (pid == 0)
    {
        /* fils2 */
        for (i = 51; i <= 100; i++)
            printf("%d\n", i);
        return 0;
    }
    wait(NULL);
    return 0;
}
```

1 point pour structure correcte

0,5 point pour type correcte de pid

2 point pour l'usage correct de fork (0 pour le fils, >0 pour le père)

2 points pour l'usage correct de wait

0,5 s'ils vérifient que fork n'a pas échoué

Ordonnancement

5. (4 points) Étant donnés les processus suivants:

<i>processus</i>	<i>T arrivée</i>	<i>Durée</i>
p1	0	6
p2	2	7
p3	3	2
p4	5	4

Donner leur ordonnancement selon l'algorithme **SRT**. Calculer le temps de rotation moyen et le temps d'attente moyen.

T

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
p 1	p 1	p1	p3	p3	p1	p1	p1	p4	p4	p4	p4	p2							

T rotation moyen:

$$((8-0) + (19-2) + (5-3) + (12-5)) / 4 = 34 / 4 = 8,5$$

T attente moyen:

$$((0+2) + (10) + (0) + (3)) / 4 = 15 / 4 = 3,75$$

2 points pour ordonnancement correct, 1 point pour T_rotation correct, 1 point pour T_attente correct