Le shell Bash

Redirection des entrées-sorties standard

Exercice 1. Donner des commandes permettant de compter : le nombre de comptes étudiants, le nombre de variables utilisées dans le shell, le nombre de fichiers dans le répertoire courant.

Exercice 2. Donner une commande permettant de créer un fichier texte contenant la ligne « bonjour à tous ». Ajouter la ligne « hello world ».

Exercice 3. Donner une commande permettant d'afficher les lignes 10 à 15 d'un fichier texte. On utilisera les commandes `head` et `tail`.

Exercice 4. Décrire rapidement ce que font les programmes `generator.c` et `filter.c`.

generator.c

```
 1 : #include <stdio.h>
 2 : #include <stdlib.h>
 3 :
 4 : int main(int argc, char *argv[])
 5 : {
 6 :     int i;
 7 :     if (argc != 2)
 8 :     {
 9 :         printf("Usage: %s <maxnum\n", argv[0]);
10 :         exit(0);
11 :     }
12 :     for (i = 2; i <= atoi(argv[1]); i++)
13 :         printf("%d\n", i);
14 :     exit(0);
15 : }
```

filter.c

```
 1 : #include <unistd.h>
 2 : #include <stdio.h>
 3 : #include <stdlib.h>
 4 :
 5 : int main(void)
 6 : {
 7 :     int p, n;
 8 :     scanf("%d", &p);
 9 :     fprintf(stderr, "[%d] %d\n", getpid(), p);
10 :     while (scanf("%d", &n) != EOF)
11 :         if (n % p)
12 :             printf("%d\n", n);
13 :     exit(0);
14 : }
```

Exercice 5. On suppose que les fichiers exécutables correspondant aux programmes ci-dessus s'appellent `generator` `filter`. Comment utiliser ces exécutables pour obtenir les dix premiers nombres premiers ?

Les scripts

Exercice 6. Écrire un script `miroir` qui affiche ses arguments dans l'ordre inverse où ils sont donnés.

Exercice 7. Écrire un script `triarg` qui affiche ses arguments dans l'ordre lexicographique. On pourra utiliser un fichier intermédiaire et la commande `sort`.

Exercice 8. Écrire un script `somme` qui fait la somme de ses arguments.

Exercice 9. Écrire un script `lsrep` qui affiche la liste des fichiers contenu dans le répertoire courant et pour chacun de ceux qui sont des répertoires en affiche le contenu.

Exercice 10. Écrire un script `totalligne` qui, pour chaque fichier source ou en-tête du répertoire courant affiche son nombre de lignes. Le nombre total de lignes est ensuite affiché.

Exercice 11. Que fait le script suivant ?

scriptx.sh

```
 1 : #!/bin/bash
 2 :
 3 : echo shell = $_
 4 : echo nom = $0
 5 : echo pid = $$
 6 : echo "nombre d'arguments = $# ($*)"
 7 : i=0
 8 : for arg in $*
 9 : do
10 :   i=‘expr $i + 1‘
11 :   echo argument $i : $arg
12 :   if cat $0 | grep $arg > /dev/null
13 :       then
14 :       echo oui
15 :       else
16 :       echo non
17 :       fi
18 : done
```

Exercice 12. Écrire un script qui donne le nombre total de fichiers contenu dans le répertoire courant et dans tous se sous-répertoires.