L2 Informatique
Correction de l'examen de Système - FLIN302
Tout document autorisé

Pierre Pompidor

vendredi 11 janvier 2008 - 8h30-10h30

Questions de cours : 6 points

Question 1.1 (paramétrages principaux du shell dans le .bashrc) :

(seuls deux paramétrages étaient demandés) - complétion de la variable d'environnement PATH - création d'alias (renommage de commandes) - modification de la valeur du prompt ...

Question 1.2 (création d'un répertoire de travail privé pour un groupe de projet) :

Dans le cadre du travail en salle de TP votre groupe principal d'appartenance n'est composé que de vous et vous n'avez pas les droits nécessaires pour changer cela, ni pour créer un groupe secondaire. Par ailleurs votre répertoire d'accueil n'est pas lisible par les autres. La solution est donc de créer un sous-répertoire de votre répertoire d'accueil (ou d'un sous-répertoire de celui-ci pour augmenter la sécurité) de droits tels que n'importe-qui pourra y écrire, mais dont vous ne donnerez le nom, (qui est en fait un mot de passe), qu'aux membres de votre projet. Commandes : mkdir MotDePasse chmod 777 MotDePasse

Question 1.3 (Pagination vs Segmentation) :

La pagination permet de pallier le manque de mémoire centrale pour l'exécution des processus en utilisant la mémoire de masse (généralement le disque dur), et cela par pages. Si la survie des processus mis ainsi en mémoire virtuelle est assuré, cela dégrade très fortement leurs temps d'exécution. La segmentation permet la répartition de la mémoire centrale en espaces d'adressage appelés segments qui vont correspondre à des parties logiques (segment de code, de données...) des processus. La gestion des segments permet également une réallocation facilitée des processus (car manipulation d'adresses relatives et non globales).

Ecriture d'un script de vérification de syntaxe python : 7 (5+2) points :

Question 2.1 :

```
#!/usr/bin/python
import sys, re

# Tableau (incomplet) des instructions à vérifier
instructions = ['for', 'while', 'if', 'else'];

fd_script_a_verifier = open(sys.argv[1], "r")
lignes = fd_script_a_verifier.readlines()
fd_script_a_verifier.close()

num_ligne = 1
```

```
for ligne in lignes :
   for instruction in instructions :
         # [^:]* = chaîne de caractères qui ne comprend pas de :
         # le second \s* ne sert pas à grand chose mais cela fait joli
         controle = re.search("^\s*"+instruction+"\s*[^:]*$", ligne)
         if controle :
             print "oubli d'un : a la ligne", num_ligne
   num_ligne += 1
```

Question 2.1 (version finale) :

```
#!/usr/bin/python
import sys, re

# Chargement dans un tableau de la liste des instructions précédant un : (for, while, if,...)
fd_instructions = open("instructions", "r")
instructions = fd_instructions.readlines()
fd_instructions.close()

# Chargement dans un tableau des lignes du script à vérifier
fd_script_a_verifier = open(sys.argv[1], "r")
lignes = fd_script_a_verifier.readlines()
fd_script_a_verifier.close()

num_ligne = 1
for ligne in lignes :
   for instruction in instructions :
       # Elimination du retour-chariot de fin de ligne
       # je n'ai pas tenu compte de cela dans la notation
       instruction = instruction[:-1]
       controle = re.search("^\s*"+instruction+"\s*[^:]*$", ligne)
       if controle :
           print "oubli d'un : à la ligne", num_ligne
    num_ligne += 1
```

Compréhension d'un script CGI : 7 points

Le script est une sorte de pendu.
Toutes les lettres à tester dans le mot qui est passé en paramètre au script (paramètre mot) apparaissent dans la page. Celles qui ont déjà été testées sont suivies de leur nombre d'occurrences dans le mot, les autres sont affichées dans un lien qui rappelle le script avec la lettre en test (paramètre test) et les lettres déjà testées (paramètres ayant pour nom le nom de la lettre). Le nombre de tentatives (qui s'incrémente à chaque rappel) est également transmis. Si le nombre de lettres trouvées est égal au nombre de lettres du mot, c'est à dire que celui-ci a été découvert !

```
Exemple :
mot = difficile
a
b
c
d
e
```

```
...
z

L'internaute clique sur f, rappel du script :
a
b
c
d
e
f : 2
...
z

et ainsi de suite.
```

Il était bon de remarquer que le fait que le mot soit transmis en paramètre en méthode GET n'était pas très judicieux, il était également agréable de proposer une solution pour "tirer" un mot (solutions comptées en bonus).