SE2 : Sujet examen 2008

Session : SE2
Promo : ING1
Durée : 1h30
Conditions : Examen rendu sur papier. Machines non autorisées, documents papiers imprimés autorisés (pas de brouillons écrits à la main en entrée de salle).
Documents annexes : short ref unix fournie.

Consignes générales :

- Lisez bien chaque énoncé avant de commencer.
- Prenez le temps d'analyser le sujet.
- N'oubliez pas la vérification des paramètres d'entrée : Une forme correcte du script peut ramener des points.
- Gardez du temps pour recopier vos scripts au propre à la fin de l'épreuve.

Exercice 1 : Déployement d'un espace de travail pour les TD

Objectif : Faire construire par un script l'ensemble des répertoires de travail pour une série de TD en langage C.

On demande interactivement le nombre de TD à mettre en place. On dispose par ailleurs dans le répertoire courant d'un makefile standard (appelé makefile). On doit créer autant de répertoires appelés exo*$n* ou *$n* est un nombre séquentiel à partir de 1.

On finit par copier le makefile dont on dispose dans les répertoires créés.

Connaissances évaluées :

Réaliser une boucle
Constituer des noms de fichier/répertoire par concaténation
Créer des répertoires
Nettoyer le système de ressources inutiles
Faire un calcul numérique (voir "Eléments de solution" de la question 5)

Rappels ou éléments de solution :

Néant. Exercice très simple.

Longueur typique : **24 lignes de code**

Durée prévue : 10 minutes

Difficulté : *

Barème : 3 points (amorce 0.5, lecture d'entrée 0.5, nettoyage préalable 0.5, fabrication des répertoires 1, copie du makefile 0.5)

Exercice 2 : Fabrication d'un index.html

Objectif : Dans un répertoire, un fichier index.html est typiquement la liste des liens Web qui permettent d'accéder à ces fichiers. Le script doit construire automatiquement un fichier index.html composé sur la base suivante :

```
<html><head></head><body><ul>
<li><a href="nomfichier">nom de base du fichier</a></li>
...
</body></html>
```

On ne doit pas prendre en compte les fichiers cachés (commencent par un '.' ou par un '!').

Connaissances évaluées :

Vérifier les paramètres d'entrée

Explorer un répertoire
Filtrer les fichiers cachés ou indésirables
Ecrire dans un fichier

Rappels ou éléments de solution :

On peut utiliser les redirections de flux standard de la commande echo pour écrire ou ajouter à un fichier.

Longueur typique : **19 lignes de code**

Durée prévue : 15 minutes

Difficulté : **

Barème : 3 points (amorce 0.5, vérification des paramètres 0.5, exploration 0.5, filtrage 0.5, ecriture 1)

Exercice 3 : Changement de casse

Objectif : baisser la casse des noms de tous les fichiers "vrais" (pas les liens ni les répertoires) d'un répertoire donné. Si le répertoire n'est pas donné, on prend le répertoire courant. On doit faire tous les tests de validité des arguments.

Connaissances évaluées :

Tester les paramètres d'entrée et savoir les compter
Opérateurs de test de fichier et de type d'entrée du SF
Parcours d'un répertoire
Extraction d'une portion de chemin pour trouver le nom local de fichier

Eléments de solution :

la commande `tr` permet de transposer une classe de caractères en une autre classe de caractères :

`tr` *ListeDépart ListeArrivée*

exemple :

```
tr àéèù aeeu
```

Elle fonctionne comme un filtre (alimentée sur l'entrée standard, produit la transformation sur la sortie standard).

Longueur typique : **23 lignes de code**

Durée prévue : 20 minutes

Difficulté : **

Barème : 4 points (amorce 0.5, tests d'entrée 1, exclusion des non-fichiers 1, fonctionnalité 1.5)

Exercice 4 : Jouez avec moi

Objectif : Un petit jeu qui tire au hazard un entier entre 1 et 20 et qui propose de le deviner. On propose d'abord de jouer par un menu Oui/Non. On doit trouver en 10 coups au maximum. Si on n'a pas trouvé dans les dix coups, on a perdu le jeu. Si on a trouvé avant les dix coups le jeu s'arrête et on a gagné. A chaque essai le programme répond si l'essai est supérieur ou inférieur au nombre choisi par le programme.

Connaissances évaluées :

menu à partir d'une boucle select
Evaluations arithmétiques et dissociation texte/arithmétique
Logique conditionnelle simple

Aides et éléments de solution :

L'évaluation numérique peut utiliser la forme de variable :

`$(( expression calculée ))`

Synonyme de l'utilisation de `expr`.

Bash fournit un générateur de nombre aléatoires sous forme d'un pseudo-variable : $RANDOM. Derrière cette variable se cache en fait un appel à la fonction C rand(). Chaque fois que la variable $RANDOM est utilisée, un nouveau nombre aléatoire entre 0 et 32767. On peut rappeler que le shell connait l'opérateur % (Modulo) dans les expressions de calcul numérique.

Longueur typique : 25 lignes de code

Durée approx : 15 minutes

Difficulté : **

Barème : 5 points (amorce 0.5, menu 1, generation 1, jeu 2, sorties de jeu 0.5)

Exercice 5 : Traverser une séquence de Turing

Objectif : On fournit une chaîne en entrée constituée de lettres majuscules.
On place un curseur (indice) à l'entrée de la chaîne. Toutes les lettres sont interprétées comme un "avance de un" sauf A, B, C, D, qui ont des significations particulières :
- A avance de 3
- B avance de 2
- C recule de 1
- D recule de 2

Le script doit exécuter le déplacement de curseur et dire si on trouve la fin
(jusqu'à un nombre de coups maximum de 150 déplacements).

Connaissances évaluées :

Extraire une lettre d'un mot (opérateur d'extraction de plage)
Savoir utiliser une meta-expression shell
Usage du case
Gérer un curseur

Eléments de solution :

On donne la formule pour extraire la lettre relativement au curseur :

```
CMD="LETTER=`echo \${INPUT:$CURSOR:1}`"
eval $CMD
```

On rappelle (encore) la façon d'obtenir un résultat numérique (une des façons de calculer).

`VAR=$(( ...expression numérique calculée... ))`

Longueur typique : 34 lignes de code

Durée approx : 20 minutes

Difficulté : ****

Barème : 5 points (amorce 0.5, tests 1, boucle à curseur 2, sélection des cas spéciaux 1.5)

Recopie et mise au propre des scripts : 15 minutes