Introduction aux Systèmes Informatiques

SCRIPT BASH

Ce TP devra être uniquement réalisée en `bash`.

Hello

1. Écrire un script qui affiche `Hello world !!!`
2. Réaliser la même opération en utilisant cette fois ci une fonction pour l'affichage.

```
#!/bin/bash

# le code de votre fonction

functionHello
```

Les variables prédéfinies

1. Modifiez votre environnement de façon à ce que les scripts inclus dans le répertoire ~/bin soit utilisable depuis n'importe quel endroit sans en spécifier le chemin d'accès. Pour ce faire vous modifierez, dans vos fichiers de configuration, la variable PATH de telle sorte que celle-ci prenne en compte votre répertoire ~/bin. Attention : la variable PATH prend peut-être déjà en compte votre répertoire ~/bin. Pour le vérifier faites echo $PATH.
2. Il est utile quand on travaille en console de différencier du premier coup d'œil le résultat d'une commande. Pour ça il vous faut modifier la valeur de la variable d'environnement PS1 (faite une echo de cette variable). Vous allez modifier votre prompt de telle sorte qu'il affiche :

   `user works on PC1 in the folder /home/user/personnal $`

   De telle manière que :

   - `user works on` soit écrit en vert
   - `PC1 in the` soit écrit en jaune
   - `folder /home/user/personnal $` soit écrit en rouge
   - `user` est le nom d'utilisateur
   - `PC1` est votre hostname
   - `/home/user/personnal` est le chemin absolu du répertoire courant

   Pour cela vous vous appuierez sur la page man de bash (dans la rubrique PROMPTING) pour ce qui est caractère spéciaux permettant de récupérer les informations nécessaires. Pour ce qui est de la couleur vous utiliserez la séquence d'échappement `\[\033[xxm\]` où `xx` est un entier. Par exemple nous pouvons changer le prompt de la manière suivante (essayez !) [1] :

```
PS1="\[\033[01;03;04;32m\]b\[\033[33m\]o\[\033[31m\]b\[\033[00m\] \$
```

Branchements conditionnels : le *if*

1. Écrire un script qui détermine si un nombre passé en argument de la commande est pair ou impair.
2. Écrire un script qui renvoie le plus grand nombre parmi les trois passés en argument de la commande.

[1] Lorsque vous appliquez un code couleur se dernier est utilisé tant que vous n'en changé par. Ainsi, ce que vous écrirez sera dans la même couleur que le dernier code couleur saisi.

Branchements conditionnels : le *case*

1. Écrire un script permettant de simuler une calculatrice pour les opérateurs +, -, *, / *et* %. Pour cela votre programme devra permettre d'effectuer une saisie du type :

```
./calc.sh 124 * 3456
```

Structures de contrôles : le *for*

1. Écrire un script qui teste si un nombre passé en argument de la commande est un nombre premier.

2. Écrire un script qui renvoie les 100 premiers nombres premier.

Structures de contrôles : le *while*

1. Écrire un script qui permet d'afficher l'ensemble des arguments passés sur la ligne de commande.

2. Écrire un script qui affiche toutes les puissances de 2, jusqu'à une valeur maximale passé en argument de la commande. Le calcul des puissances de 2 sera effectué par multiplications successives.

Gestion d'une pile en shell

Une pile est une structure de données fondée sur le principe « dernier arrivé, premier sorti » ce qui veut dire que les derniers éléments ajoutés à la pile seront les premiers à être récupérés (une pile d'assiettes).

Exercice : Proposer un script *shell* qui implémente les fonctions suivantes :

```
#!/bin/bash

# Push an element
# $1, the stack
# $2, the element to add
function push
{
}

# Pop an element
# $1, the stack
function pop
{
}


# Print the top of the stack
# $1, the stack
function top
{
}
```

Calculatrice notation polonaise inverse

La notation polonaise inverse (NPI), également connue sous le nom de notation post-fixée, permet de noter les formules arithmétiques sans utiliser de parenthèses.
Considérons pas exemple l'expresion suivante :

7 * 2 + 3

s'écrira :

7 2 * 3 +

L'évaluation d'une expression écrite en post-fixée se base sur l'utilisation d'une pile; c'est-à-dire que les opérandes sont ajoutés en haut de la pile, et les résultats des calculs sont retournés en haut de la pile. Sur l'exemple on a :

a. empiler 7

b. empiler 2

c. dépiler les deux opérandes, effectuer le calcul et ajouter le résultat sur la pile

d. empiler 3

e. dépiler les deux opérandes, effectuer le calcul et ajouter le résultat sur la pile

Exercice : Écrire un script shell permettant d'évaluer une expression écrite en notation polonaise inverse.

Exercices divers

1. Un entier strictement positif x est un nombre factoriel s'il existe un autre entier y tel que $x = y!$. Écrire un script qui étant donné un entier x, vérifie si x est un nombre factoriel, et, si c'est le cas affiche la valeur de y. L'entier x est passé en argument de la commande.

2. Écrire un script permettant de déterminer le nombre de voyelles (a, e, i, o, u et y) d'un mot passé en argument de la commande.

3. Étant donné un entier n fourni en argument de la commande, écrire un script permettant d'afficher une pyramide constituée de n lignes de lettres comme l'illuste l'exemple suivant (avec $n = 5$) :

```
        a
      a b a
    a b c b a
  a b c d c b a
a b c d e d c b a
```