Travaux Pratiques (TP)
La programmation avec le Shell/Unix

Ce TP fait l'objet d'une notation. Vous travaillerez seul et vous devrez rendre un compte-rendu. Les TP sont à remettre **avant le vendredi 13 novembre 2020** *sur EDUNAO. Vous devez envoyer un fichier au format PDF qui devrait être nommé avec la convention suivante : NOM-TP2.pdf.*

Introduction

On rappelle qu'un script shell est un fichier texte d'extension `.sh`, comme `toto.sh`. Un tel script peut être exécuté en commençant le script par une ligne précisant l'interpréteur :

#!/bin/bash

Le fichier est ensuite rendu exécutable par la commande

```
chmod +x toto.sh
```

et puis exécuté par

```
./toto.sh.
```

Exercice 1 : Sauvegarde automatique de fichiers

- Le but de cet exercice consiste à définir un script permettant de sauvegarder tous les fichiers du répertoire courant dans un autre répertoire avec, en plus, une indication du jour où la sauvegarde est effectuée.
- Ecrire un script bash copiant tous les fichiers du répertoire courant dans le sous-répertoire `OLD` en ajoutant au nom des fichiers la date du jour au format `"#année-mois-jour".` Il faudra, au préalable, s'assurer de l'existence du répertoire `OLD` et le créer s'il n'existe pas. On vérifiera alors dans ce cas que la création du répertoire s'est bien passée.
- Par exemple, si le répertoire courant comporte un fichier `fich` et que le shell est lancé aujourd'hui, ce programme devra recopier le fichier `fich` vers le fichier `OLD/fich#2020-11-04`.

Exercice 2 : Gestion d'une Corbeil de fichiers

- Ecrire un script `jeter.sh` qui permet de manipuler une poubelle à fichier (un répertoire nommé `poubelle` situé à votre racine).
  ```
  ./jeter.sh fichier1 fichier2 ...
  ```
  pour déplacer les fichiers considérés vers la poubelle ;
- La commande accepte trois options :
  1. `-l` pour lister le contenu de la poubelle ;
  2. `-s` fichier chemin pour sortir le fichier de la poubelle et le placer 'a l'em- placement chemin ;

3. `-v` pour vider la poubelle ;

— Si la poubelle n'existe pas, elle est créée à l'appel de la commande.

Exercice 3 : Compilation de fichiers Java

— *Pour réaliser ce TP, vous devez installer Java sur votre machine en suivant le tutoriel disponible sur EDUNAO. Vous devez également télécharger les ressources de l'exercice 3 - TP2 sur EDUNAO.*

— Définir un script shell `javac_test.sh` qui attend un argument et teste si cet argument est un fichier **Java** compilable sans erreurs.
  — Si l'argument est un fichier **Java** qui se compile sans aucune erreur, il affiche sur la sortie standard `file.java : compilé`, génère un fichier `file.class`, et termine avec le code de retour 0.
  — Si l'argument est un fichier **Java** qui ne se compile pas, il affiche sur la sortie standard `file.java : non compilé` et termine avec le code de retour 1.
  — Dans tous les autres cas, il affiche `Entree incorrecte` sur la sortie d'erreur standard et termine avec le code de retour 2.

  Pour compiler un fichier **Java**, il faut exécuter la commande :

```
javac fichier.java
```

— Définir un script shell `java_compilation.sh` qui prend comme argument un répertoire `rep` qui contient des fichiers (**Java** ou autre), parcours le contenu du répertoire, exécute `javac_test.sh`, et sauvegarde les fichiers `.class` générés dans un répertoire `rep/bin`.

— Définir un script shell `java_execution.sh` qui prend comme argument un répertoire `rep`, teste si le répertoire `rep/bin` existe et exécute les fichiers `.class` générés par le script de la question précédente.

  Pour rappel, seul les fichiers **Java** contenant une méthode avec la signature `public static void main` peuvent être exécutés.

  Pour exécuter un fichier **Java**, il faut exécuter la commande :

```
java fichier
```

Vous pouvez utiliser les ressources téléchargées depuis EDUNAO pour tester votre script.