Projet

Instructions

Merci de lire attentivement les instructions avant de commencer le projet.

Date de remise : **17/12/2023 à 23h59**. Toute heure de retard enlèvera 1 point sur la note finale.

Projet par groupes de 2 ou 3. Internet autorisé. Cours autorisé. **Intelligence artificielle interdite.** Merci de ne pas communiquer entre groupes.

Travail à rendre

Ce projet contient 3 exercices indépendants centrés autour d'un problème. Chaque exercice consiste en l'écriture d'un script bash :

`exo1.sh`, `exo2.sh`, `exo3.sh`

Créez un **dossier** dont le nom est la suite des noms de famille (avec des tirets à la place des espaces) des membres du groupes, séparés par des underscores. Exemple:

`Richard_Lefebvre_de-Duras`

C'est dans ce dossier que vous devrez créer les scripts de ce projet.

Le 17 décembre, vous devez **compresser ce dossier au format zip ou tar.gz ou tar.xz** et le rendre sur Arche. Pour résumer, je dois donc recevoir de vous un fichier `noms.zip` (ou `noms.tar.gz` out `noms.tar.xz`) (où `noms` est vos noms de famille) qui contient `exo1.sh`, `exo2.sh` et `exo3.sh`. Tous les autres fichiers de ce dossier seront ignorés.

Commande pour créer une archive tar à partir d'un dossier:

```
$ tar -czvf nom_du_dossier.tar.gz nom_du_dossier
```

Vérifiez que le contenu n'est pas vide avec

```
$ tar -tvf nom_du_dossier.tar.gz
```

Attention : Si votre rendu contient un fichier mal nommé ou si l'archive n'est pas au bon format, un point sera retiré à la note finale.

Mise en garde et notation

Attention : Dans ce TP, il faut référer aux paramètres des scripts (ex. le premier paramètre est `$1`).

Note : Les messages d'erreur sont impérativement à envoyer dans la sortie erreur. Attention aussi à ne pas laisser de messages d'erreurs dus aux erreurs de syntaxe.

Pour corriger votre travail, j'utiliserai un script qui teste chaque question et vérifie si votre script renvoie la bonne réponse, sur une entrée qui n'est pas indiquée dans ce devoir.

Faites donc bien attention aux consignes précises de chaque question.

Chaque question de chaque exercice rapport 1 point sur 15. Pour chaque exercice, 1 point supplémentaire sera attribué en regardant le code source. Il sera notamment attribué si le résultat n'est pas exactement bon ou si le code ne fonctionne pas à cause d'une erreur de syntaxe, mais que les idées sont présentes dans le code. Il sera aussi attribué si le code est compréhensible et lisible.

Merci donc de commenter un minimum votre code (avec `#`).

Des exemples d'utilisation sont proposés. **Votre code doit marcher sur ces exemples** et renvoyer exactement ce qui est demandé, au caractère près. Faites attention à retester votre code sur les premières questions après avoir finit chaque exercice.

Note : L'exercice 2 est un peu plus difficile que l'exercice 3.

Exercice 1

1) Créez un script `exo1.sh` qui affiche son deuxième paramètre. Exemple :

```
$ exo1.sh le monde est beau
monde
```

2) Modifiez `exo1.sh` pour qu'il affiche en fait juste le troisième caractère de son deuxième paramètre. Exemple :

```
$ exo1.sh le monde est beau
n
```

3) Modifiez `exo1.sh` pour qu'il affiche en plus, sur une autre ligne, le nom d'utilisateur de la personne qui lance `exo1.sh`. Exemple :

```
varichar@valpro:~$ exo1.sh le monde est beau
n
varichar
```

4) Modifiez `exo1.sh` pour qu'il affiche ces deux lignes dans la sortie erreur.

Exercice 2

Dans cet exercice, on utilise un fichier `csv` qui recense des informations anonymisées sur des joueuses et joueurs de foot professionnels. Le fichier comporte 6 colonnes :

1. salaire à l'année (en euros)
2. âge
3. club
4. ligue
5. pays

6. poste

Vous pouvez tester votre script avec le fichier `salaire.csv` sur Arche. Voici un extrait de `salaire.csv` :

```
2871000,27,Brianza,Serie A,ITA,Forward
1293000,25,Getafe,La Liga,PAR,Defender
2149000,30,OL,Ligue 1 Uber Eats,ARG,Defender
5200000,29,Tottenham,Premier League,WAL,Defender
7850,23,PTM,Primiera Liga,BRA,Defender
```

J'évaluerai votre script avec un autre fichier, qui a la même structure mais des données différentes.

1) Créez un script `exo2.sh` qui affiche le contenu du fichier donné en paramètre, mais en montrant seulement les défenseurs (`Defender`) de la Bundesliga. Exemple :

```
$ ./exo2.sh salaire.csv | head -n 5
2149000,32,FC Augsburg,Bundesliga,GER,Defender
2017000,28,SC Freiburg,Bundesliga,GER,Defender
2900,18,VfB Stuttgart,Bundesliga,GER,Defender
2579000,27,Borussia M'gladbach,Bundesliga,ALG,Defender
8598000,33,Borussia Dortmund,Bundesliga,GER,Defender
```

2) Modifiez `exo2.sh` pour qu'il n'affiche en fait que le salaire et le club.

(Indication : On pourra utiliser la commande ***`cut`****)*

Exemple :

```
$ ./exo2.sh salaire.csv | head -n 5
2149000,FC Augsburg
2017000,SC Freiburg
2900,VfB Stuttgart
2579000,Borussia M'gladbach
8598000,Borussia Dortmund
```

3) Modifiez `exo2.sh` pour que le résultat affiché soit trié par ordre croissant sur l'âge des joueurs et joueuses. Exemple :

```
$ ./exo2.sh salaire.csv | head -n 5
1325,FC Augsburg
2100,VfB Stuttgart
2900,Hertha Berlin
2900,Hoffenheim
2900,VfB Stuttgart
```

4) (*Difficile*) Modifiez `exo2.sh` pour que le nom du club soit réduit au dernier mot (ex. `Borussia M'gladbach` → `M'gladbach`), sauf quand le deuxième mot est une suite de chiffres non vide, auquel cas on garde le premier mot (ex. `Bayer 04` → `Bayer`).

Exemple :

```
$ ./exo2.sh salaire.csv | head -n 10
1325,Augsburg
2100,Stuttgart
2900,Berlin
2900,Hoffenheim
2900,Stuttgart
2900,Stuttgart
2900,Bochum
3750,Bochum
4300,Schalke
5150,Mainz
```

Exercices 3

Situation

L'entreprise CriSpam commercialise une intelligence artificielle entrainée à reconnaitre les courriels indésirables. Pour évaluer la performance de leur programme, elle la teste sur une liste d'email dont on si s'ils sont des spams ou pas.

Les emails sont classés en 4 catégories : `TP`, `FP`, `TN` et `FN`.

Le score F1 de l'IA est calculée par la formule suivante :

$$F1 = \frac{2\texttt{TP}}{2\texttt{TP} + \texttt{FN} + \texttt{FP}}$$

où `TP` signifie le nombre d'email appartenant à la catégorie `TP`, etc.

Questions

1) Créez un script `exo3.sh` qui prend en paramètre quatre nombres : `TP`, `FP`, `TN` et `FN`, et affiche le score F1 de l'IA. Exemple :

```
$ exo3.sh 90 10 80 20
.85714285714285714285
```

2) Modifiez `exo3.sh` pour que, si on ne lui donne pas exactement 4 paramètres, il renvoie un message d'erreur et s'arrête avec un code d'erreur de 3.

3) Modifiez `exo3.sh` pour que, si l'option `--help` est donné seule, le script affiche un descriptif de l'utilisation de lui-même et renvoie un code de retour de 0. Le descriptif n'est pas imposé mais je vous conseille celui-ci :

```
$ exo3.sh --help
Usage: exo3.sh TP FP TN FN
Computes the F1 score
```

4) (*Plus dur*) Modifiez `exo3.sh` pour que, si le score F1 est en dessous de 0,8, `exo3.sh` affiche en plus (et sur une autre ligne) un message d'erreur. Exemple :

```
$ exo3.sh 70 30 80 20
.73684210526315789473
Attention, score F1 bas !
```