TP - Bash

Votre mission

En utilisant les fichiers de logs fournis

- Lister les requêtes les plus fréquentes
- Blacklister les IP de requêtes malicieuses
- Générer une liste des mise online/offline du serveur

Ce TP est divisé en étapes digestes, utilisez un maximum les outils d'aides pour tous les éléments que vous ne savez pas (*encore*) faire

Validez chaque étape avant de commencer l'étape d'après

Pour vous aider:

- `<commande> --help` ou `<commande> -h`
- `man <commande>`
- Stack Overflow
- Google
- Moi

1 - Récupérer les arguments

Créer une condition pour s'assurer que **3** arguments ont été renseignés

Si non, stopper l'exécution en renvoyant le code d'erreur `1` et afficher un message:

```
Usage: $0 <actix_log> <nginx_access_log> <nginx_error_log>
```

2 - Initialiser les variables

Pour chaque argument, initialiser une variable avec le *chemin absolu* du fichier passé en paramètre

- **ACTIX_LOG** récupère le chemin absolu de l'argument numéro **1**
- **NGINX_ACCESS_LOG** récupère le chemin absolu de l'argument numéro **2**
- **NGINX_ERROR_LOG** récupère le chemin absolu de l'argument numéro **3**

3 - Les requêtes les plus fréquentes

Dans **ACTIX_LOG**, récupérer toutes les requêtes `GET` avec un status `200`

Parmis ces requêtes, n'afficher que le chemin (ex: `/index.html`).
Note: Il s'agit du 7e élément de la ligne de log

Trier ces lignes par ordre alphabétique, puis compter le nombre de ligne unique présente

Trier le résultat par le nombre d'occurence (indiqué dans la 1e colonne)

Enregistrer ce résultat dans une variable **MOST_SERVED**

4 - Filtrer les requêtes

Boucler à travers une liste d'extensions à filtrer: *png, css, ico et js*

```
LIST=("a" "b" "c" "d")              # Sans virgule !
for element in "${LIST[@]}"; do     # Oui c'est ignoble
  # ...
done
```

Pour chaque filtre, retirer de **MOST_SERVED** les lignes possédant cette extension
Attention à ne pas retirer d'autres requêtes ! (ex: /p **ico** n/biere)

Écraser la valeur de **MOST_SERVED** avec cette version filtrée

5 - Ne garder que les meilleurs

Parmis les requêtes restantes, ne garder que celles qui ont des occurences `> 10`

Formatter la ligne en `/<path> : <count>`

Enregistrer le résultat dans un fichier `most_served.txt`

6 - IP Blacklist

Dans **NGINX_ACCESS_LOG**, récupérer les requêtes aux endpoints: (case insensitive)
- admin
- debug
- login
- .git

Ajouter les IP de toutes ces requêtes dans un fichier `ip_blacklist.txt`

7 - Encore plus de blacklist

Ajoutez à `ip_blacklist.txt` les IP des requêtes ayant une méthode qui ne soit pas:

- GET
- POST
- HEAD

Trier les IP, et retirer tous les doublons du fichier

8 - Repérer les downtimes

Downtime = Le moment `nginx` n'arrive pas à dialoguer avec `actix_web`

Dans le fichier **NGINX_ERROR_LOG**, récupérer les erreurs du type:
`connect() failed (111: Unknown error) while connecting to upstream`

Récupérer la date et l'heure de chacune de ces erreurs

Formater les données au format: `<date> <heure> DOWN`

Enregistrer le résultat dans une variable **DOWNTIME**

9 - Récupérer les uptimes

À partir des regex:

```
DATE_REGEX="([0-9]{4})-([0-9]{2})-([0-9]{2})"
TIME_REGEX="([0-9]{2}):([0-9]{2}):([0-9]{2})"
```

Récupérer les lignes de **ACTIX_LOG**, et afficher uniquement la date:

- Chaque groupe `(...)` du regex est un *capture group*
- Le *capture group* numéro N peut être utilisé dans le format avec `\N`
- Avec un format `YYYY/MM/DD H:M:S UP`
- En utilisant la commande `sed -nr "s+<votre regex ici> .*+<format>+p"`

Enregistrer les résultats de cette commande dans une variable **UPTIME**

10 - Gros travail awk

En affichant données avec `echo -e "$DOWNTIME\n$UPTIME"`:

- Trier les entrées par date, puis par heure

Puis créer un algorithme avec `awk` pour chaque ligne:

- Sachant que l'état initial est "DOWN"
- Si la ligne indique "UP" et l'état précédent était "DOWN":
  - Afficher la date, l'heure et le message "UP"
  - L'état est maintenant "UP"
- Si la ligne indique "DOWN" et l'état précédent était "UP":
  - Afficher la date, l'heure et le message "DOWN"
  - L'état est maintenant "DOWN"

```
awk -vma_variable="toto" '
  # $NF = dernier élément de la ligne
  $NF == "a" && ma_variable == "tutu" {
    ma_variable = "toto"
    print "message"
  }

  ma_variable == "toto" {
    ma_variable = "tutu"
    print $1 " " $2 "toto"
  }
'
```

Enregistrer le résultat dans un fichier `server_status.txt`