

Optimisation d'une commande Bash

<https://gemini.google.com/app/751425cdf1134d8d>

User prompt: mieux coder cette ligne : `dist="$(cat /etc/os-release | grep 'PRETTY_NAME' | sed -r 's/^.{13}/' | sed -r 's/'/'/'); mes="$(awk '{print " a démarré depuis "int($1/3600)" heure(s), "int(($1%3600)/60)" minute(s)s, et "int($1%60)", "int(($1%60)/10)" seconde(s)."}' /proc/uptime)"; echo $dist $mes`

Response: On peut grandement simplifier et optimiser cette ligne de commande :

```
dist=$(sed -n 's/^PRETTY_NAME="\(.*\)"/\1/p' /etc/os-release)
mes=$(awk '{h=int($1/3600); m=int(($1%3600)/60); s=int($1%60); printf "a démarré depuis %d heure(s), %d minute(s), et %d seconde(s).", h, m, s}' /proc/uptime)
echo "$dist $mes"
```

Ce qui a été amélioré

1. Extraction de l'OS (dist) :

- **Inutile de multiplier les sous-processus (cat, grep, sed)** : un seul sed suffit pour lire le fichier, chercher la ligne PRETTY_NAME et capturer le contenu entre guillemets.
- Remplacement de la syntaxe obsolète des backticks ``...`` par `$(...)`.

2. Calcul de la durée de fonctionnement (mes) :

- **Corrections typographiques** : suppression des fautes (`minute(s)s` → `minute(s)`).
- **Calcul des secondes** : suppression du calcul erroné des décimales de secondes (`"int($1%60)", "int(($1%60)/10)"`).
- **Clarté avec printf** : utilisation de variables intermédiaires et de spécificateurs de format `%d` pour éviter d'échapper inutilement les parenthèses (`\(`).

3. Bonnes pratiques Bash :

- Ajout de guillemets autour de `"$dist $mes"` lors du echo pour éviter le *word splitting* (découpage inattendu des espaces).

Alternative condensée (tout dans AWK)

Si l'objectif est d'avoir une seule commande rapide à exécuter dans un script ou un terminal :

```
awk -F=' ' '/PRETTY_NAME/{gsub("/", "", $2); dist=$2} END {getline < "/proc/uptime"; h=int($1/3600); m=int(($1%3600)/60); s=int($1%60); printf "%s a démarré d
```
