TP : Surveillance de Connectivité Réseau

Créer un script en Python de surveillance de connectivité réseau. Ce script devra être capable de lire une liste d'hôtes à partir d'un fichier CSV, de tester leur accessibilité via des commandes de ping, et d'enregistrer les résultats dans un fichier de log. En cas de défaillance de connexion, les hôtes inaccessibles seront listés dans un fichier séparé.

Contexte

Dans le cadre de l'administration réseau, la surveillance de la disponibilité des serveurs et des équipements est essentielle. En automatisant ce type de vérification, vous serez en mesure d'identifier rapidement des problèmes de connectivité et de documenter les pannes.

Objectifs de l'exercice

1. **Apprendre à utiliser les commandes système avec Python** pour effectuer des tests de connectivité réseau.
2. **Lire et traiter un fichier CSV** pour extraire des informations réseau.
3. **Gérer l'enregistrement des résultats dans des fichiers de log**.
4. **Modulariser le script** en définissant des fonctions pour chaque tâche.

Avant d'écrire le code, vous devez structurer votre raisonnement sous la forme d'un **logigramme** ou d'un **algorithme en pseudo-code**. Cette étape est essentielle pour clarifier les étapes du programme et organiser la logique de traitement.

Le logigramme doit :

1. Décrire l'enchaînement des opérations.
2. Identifier les variables et les fonctions clés (ex. ping_host, log_result, etc.).
3. Prévoir les étapes de contrôle de flux (ex. vérification de la connectivité, gestion des échecs).

En utilisant votre logigramme ou pseudo-code comme guide, vous allez écrire le code Python correspondant pour exécuter la surveillance réseau. Voici les spécifications et étapes pour le développement du code Python.

1. Étape 1 : Ping d'un Hôte

- **Objectif :** Apprendre à effectuer un ping d'un hôte pour vérifier sa disponibilité.
- **Instructions :**
    - Créer une fonction **ping_host(host)** qui prend une adresse IP ou un nom de domaine comme argument et vérifie s'il est accessible.
    - Renvoyer **True** si l'hôte est accessible, False sinon.
    - **Conseil :** Utiliser **os.system** pour exécuter une commande ping en fonction du système d'exploitation (Windows ou Linux/Mac).
- **Éléments à évaluer :** Capacité à adapter la commande de ping en fonction de l'OS.

2. Étape 2 : Enregistrement des Résultats

- **Objectif :** Enregistrer les résultats du ping dans un fichier de log.
- **Instructions :**
    - Créer une fonction **log_result(host, status, log_file)** qui prend en paramètres l'hôte, son statut (Accessible/Inaccessible) et le nom d'un fichier de log.
    - Ajouter un enregistrement dans le fichier de log avec la date et l'heure, le nom de l'hôte et son statut.
- **Éléments à évaluer :** Utilisation de **with open()** et de l'option d'écriture, formatage du timestamp.

3. Étape 3 : Lecture d'un Fichier CSV d'Hôtes

- **Objectif :** Charger une liste d'hôtes à partir d'un fichier CSV. Le script doit lire un fichier CSV contenant une liste d'hôtes à surveiller (adresse IP ou nom de domaine).
- **Instructions :**
    - Utiliser la bibliothèque **csv** pour lire un fichier contenant une liste d'adresses IP ou de noms de domaines, avec une adresse par ligne.
    - Boucler sur chaque hôte, utiliser **ping_host()** pour vérifier sa disponibilité, et enregistrer chaque résultat avec **log_result()**.
- **Éléments à évaluer :** Capacité à lire un fichier CSV et à traiter chaque ligne.

4. Étape 4 : Enregistrement des Échecs de Connexion

- **Objectif :** Enregistrer les hôtes inaccessibles dans un fichier séparé pour les échecs.
- **Instructions :**
    - Ajouter une fonction **log_failures(failures, log_file)** qui enregistre la liste des hôtes inaccessibles avec un timestamp dans un fichier distinct.
    - Dans la boucle principale, conserver les hôtes inaccessibles dans une liste failures, puis appeler **log_failures()** après la boucle.
- **Éléments à évaluer :** Manipulation des listes et condition de vérification pour les hôtes inaccessibles.

5. Étape 5 : Structure et Modularisation du Script Principal

- **Objectif :** Organiser le script en une fonction principale.
- **Instructions :**
    - Créer une fonction **main()** qui lit le fichier CSV, effectue les tests de ping, enregistre les résultats et les échecs.
    - Gérer les erreurs (ex. vérifier l'existence du fichier CSV avant d'essayer de l'ouvrir).

- **Éléments à évaluer :** Compréhension de la structure d'un script, utilisation de la fonction **main()** pour organiser le flux du programme.

6. Étape 6 : Utilisation des Arguments en Ligne de Commande

- **Objectif :** Rendre le script plus flexible en permettant de passer le nom du fichier CSV en argument.
- **Instructions :**
    - Utiliser le module sys pour récupérer le nom du fichier CSV à partir des arguments en ligne de commande.
    - Ajouter un message d'erreur si aucun fichier n'est fourni en argument.
- **Éléments à évaluer :** Utilisation de sys.argv pour les arguments de ligne de commande et gestion des erreurs.

7. Étape Avancée : Ajout de l'Intervalle de Vérification (Optionnel)

- **Objectif :** Automatiser les tests de ping avec un intervalle défini.
- **Instructions :**
    - Ajouter une boucle while pour répéter la vérification des hôtes toutes les X minutes (définies par l'utilisateur).
    - Attendre l'intervalle défini en utilisant time.sleep().
- **Éléments à évaluer :** Boucles infinies contrôlées, gestion de l'intervalle de temps.