L3 TP AIS — PYTHON

Séance 3 · Fonctions et fichiers

Fil rouge : produire un rapport d'audit · Durée indicative : 2 h

Du script au petit outil

Le script de la séance 2 mélangeait lecture des données, décision et affichage. Nous allons organiser ce travail : une fonction réalise une tâche précise ; un fichier fournit l'inventaire ; le programme produit un rapport.

Séance 1	Séance 2	Séance 3
un serveur	un parc	un outil organisé

Objectifs

- définir et appeler une fonction ;
- utiliser paramètres et valeur de retour ;
- lire un fichier texte ligne par ligne ;
- traiter un inventaire simple ;
- écrire un rapport d'audit.

Fil rouge — Chaque ligne du fichier décrira un serveur et sa charge. Notre fonction attribuera un verdict.

1. Définir une fonction

Une fonction regroupe des instructions sous un nom. Elle est définie avec def, puis appelée plus loin dans le programme.

```
def afficher_bonjour():
    print("Bonjour AIS")

afficher_bonjour()
```

Pourquoi ?

- éviter de recopier le même code ;
- donner un nom clair à une action ;
- faciliter la lecture, les tests et les modifications.

Indentation — Le bloc de la fonction est indenté. Définir une fonction ne l'exécute pas : il faut l'appeler.

2. Paramètres et retour

Un paramètre permet de fournir une donnée à la fonction. return renvoie un résultat au programme appelant.

```
def audit_charge(charge):
    if charge > 0.85:
        return "ALERTE"
    return "OK"

verdict = audit_charge(0.92)
print(verdict)
```

Élément	Rôle
charge	paramètre reçu par la fonction
"ALERTE" / "OK"	valeur renvoyée par return
verdict	variable qui récupère le résultat

Exercice rapide

Écrivez une fonction bonjour(nom) qui affiche « Bonjour », suivi du nom fourni.

À distinguer — print affiche ; return transmet une valeur pour la réutiliser ailleurs.

3. Lire un fichier texte

Un fichier permet de séparer les données du programme. Ici, inventaire.txt contient une ligne par serveur :

```
web-01;0.35
db-01;0.92
api-01;0.61
cache-01;0.88
```

Ouvrir et parcourir

```
with open("inventaire.txt", "r") as fichier:
    for ligne in fichier:
        print(ligne.strip())
```

strip() retire le retour à la ligne final. Le mode "r" signifie lecture (read).

Organisation — Placez inventaire.txt dans le même dossier que votre script, sauf consigne contraire.

4. Découper une ligne

split(";") découpe un texte au niveau du séparateur ; les valeurs obtenues restent du texte et doivent être converties si nécessaire.

```
ligne = "db-01;0.92"
elements = ligne.strip().split(";")
nom = elements[0]
charge = float(elements[1])
```

Variable	Valeur
nom	"db-01"
charge	0.92
type(charge)	float

Exercice rapide

Avec la ligne "web-01;0.35", affichez seulement le nom puis seulement la charge.

Piège courant — float doit recevoir "0.92", pas toute la ligne "db-01;0.92".

5. TP guidé · Rapport d’audit

Créez audit_fichier.py et inventaire.txt. Le programme doit lire chaque serveur, appeler une fonction d’audit et afficher un bilan.

Étape 1 — Écrire la fonction

Créer audit_charge(charge) qui renvoie « ALERTE » si la charge est strictement supérieure à 0.85, sinon « OK ».

Étape 2 — Lire le fichier

1. Ouvrir inventaire.txt avec with open(...).
2. Parcourir chaque ligne avec for.
3. Découper la ligne avec strip().split(";").
4. Récupérer nom et charge (float).

TP guidé · suite

Étape 3 — Produire une ligne de rapport

Pour chaque serveur, appeler audit_charge(charge), puis afficher par exemple : « db-01 : 92 % — ALERTE ».

Étape 4 — Bilan

5. Créer nb_serveurs et nb_alertes avant la boucle.
6. Augmenter nb_serveurs à chaque ligne.
7. Augmenter nb_alertes lorsque le verdict vaut "ALERTE".
8. Afficher le bilan après la lecture du fichier.

Attendu	Valeur
serveurs lus	4
alertes	2
machines concernées	db-01, cache-01

Indice — Le résultat de la fonction peut être stocké : verdict = audit_charge(charge).

6. Défi · Écrire un rapport

Au lieu d'afficher seulement le rapport, créez rapport.txt. Une ligne doit être écrite pour chaque serveur, puis une ligne de bilan.

```
with open("rapport.txt", "w") as rapport:
    rapport.write("Début du rapport\n")
```

Bonus

- Ajouter la date (facultatif, après recherche collective).
- Ajouter une alerte si une ligne est vide.
- Conserver une liste des serveurs en alerte.

Attention — Le mode "w" crée ou remplace le contenu d'un fichier. Testez sur rapport.txt, jamais sur inventaire.txt.

7. Travail personnel

Exercice 1 — Fonction RAM

Écrire audit_ram(ram_go) : renvoyer « RAM insuffisante » sous 8, sinon « RAM OK ».

Exercice 2 — Fichier de services

Créer services.txt avec un service par ligne. Lire et afficher « Vérification de … » pour chaque ligne.

Exercice 3 — Rapport disque

À partir de nom;espace_libre, produire un verdict « critique » si l'espace libre est inférieur à 15.

Débogage

```
def audit_charge(charge)
    if charge > 0.85:
        return ALERTE
    else:
        print("OK")

charge = float("db-01;0.92")
```

Pour la suite — Vous savez maintenant séparer données, traitement et rapport : la base d'une automatisation plus robuste.