Contrôle continu n° 1 : Pratique

1 – Devine le modèle de voiture

Énoncé :
Le programme choisit **un modèle de voiture secret** (exemple : “tesla”) à partir **d’un fichier texte** contenant une liste de modèles (par exemple `modeles.txt`).
Le joueur doit **deviner les lettres une par une**.
Chaque erreur fait **perdre un point de vie**.

Exemple d’affichage :

```
Modèle : _ _ _ _ _
Essais restants : 6
Lettre ? a
Modèle : _ e _ _ a
```

- **Fonctionnalités :**

1. **Lecture des modèles** :
   - Le programme lit les modèles de voitures à partir du fichier `modeles.txt`.
   - Il en choisit un **au hasard**.
2. **Déroulement du jeu** :
   - Le joueur propose des lettres une par une.
   - Chaque erreur retire un point de vie.
   - Le programme affiche les lettres déjà proposées et l’état actuel du mot.
3. **Enregistrement des résultats** :
   - À la fin du jeu, le programme **enregistre le résultat** (nom du joueur, mot trouvé, nombre d’essais utilisés, victoire ou défaite) dans un fichier `scores.csv`.

- **Exemple de fichiers utilisés :**

- `modeles.txt`

```
tesla
audi
toyota
ford
bmw
renault
peugeot
mercedes
```

- `scores.csv`

```
joueur,mot,trouvé,essais_restants,résultat
Abderrahman,tesla,oui,2,gagné
Ali,bmw,non,0,perdu
```

2 – Détecteur d'insultes

Objectif :

Créer un programme Python capable de **détecter et masquer automatiquement** les mots considérés comme "malpolis" dans un message texte, en utilisant les **expressions régulières** (`re`).

Énoncé :

On souhaite développer un petit outil appelé **"Détecteur d'insultes"**.
Cet outil recevra un texte saisi par l'utilisateur (par exemple un message ou un commentaire) et devra :

1️⃣ Identifier les **mots interdits** présents dans le texte (parmi une liste prédéfinie).

```
# Exemple de mots interdis
mots_interdits = [
    "bête", "idiot", "stupide", "nul", "imbécile", "abruti", "incapable",
    "fainéant", "méchant", "agaçant", "insolent", "prétentieux", "paresseux",
"menteur",
    "boulet", "clown", "rigolo", "maladroit", "distrait"
]
```

2️⃣ Remplacer chaque mot interdit par **** afin d'anonymiser ou de filtrer le message.
3️⃣ Afficher le texte filtré à l'écran.

👉 Le filtrage doit être **insensible à la casse** (par exemple, "BÊTE" ou "bête" doivent être reconnus).
👉 Les mots doivent être détectés **même s'ils sont entourés de ponctuation**.
👉 Tu dois utiliser les **expressions régulières** pour effectuer la détection.

Bonus

1. Sauvegarde les **mots interdits détectés** dans un fichier texte nommé `insultes_detectees.txt`.
2. Ajoute une fonctionnalité qui lit la **liste des mots interdits depuis un fichier JSON** ou `txt` au lieu de les écrire directement dans le code.

Exemple d'exécution :

```
Entrée utilisateur : "Tu es vraiment bête et insolent, mais un peu rigolo !"

Texte filtré : "Tu es vraiment **** et ****, mais un peu **** !"
```

---

3 – Gestion d'un tournoi sportif :

Objectif

Créer une application Python qui simule un tournoi de football **avec sauvegarde et chargement des données** depuis des fichiers (`JSON`, `CSV`, `TXT`).

Énoncé

Tu dois concevoir un programme qui gère un petit tournoi de football entre plusieurs équipes.
Chaque équipe possède les informations suivantes :

```
{
  "nom": "IA104",
  "points": 0,
  "buts_marques": 0,
  "buts_encaisses": 0
}
```

Fonctionnalités demandées

1 Gestion des équipes

- Ajouter une ou plusieurs équipes participantes.
- Sauvegarder la liste des équipes dans un **fichier JSON** (`equipes.json`).
- Pouvoir **charger** la liste d'équipes depuis le fichier pour reprendre un tournoi existant.

2 Gestion des matchs

- Enregistrer le résultat d'un match au format :
- `IA104 2 - 1 IA103`
- Le programme doit :
  - Mettre à jour les buts marqués et encaissés pour chaque équipe.
  - Mettre à jour les points selon les règles :
    - victoire = +3 points
    - nul = +1 point
    - défaite = 0 point
- Chaque match doit être **enregistré dans un fichier texte (`matchs.txt`)** pour garder l'historique.

3 Classement général

- Calculer et **afficher le classement général** :
  - trié d'abord par **points décroissants**
  - puis par **différence de buts** (`buts_marques - buts_encaisses`)
- Le classement final doit être exporté dans un **fichier CSV (`classement.csv`)** contenant les colonnes :

```
Nom, Points, Buts marqués, Buts encaissés, Différence
```

Statistiques du tournoi

- Afficher :
  - La meilleure attaque (équipe avec le plus de buts marqués)
  - La meilleure défense (équipe avec le moins de buts encaissés)
- Ces informations peuvent aussi être écrites dans un petit **fichier texte (`stats.txt`)**.

Fichiers à manipuler

Fichier	Format	Contenu
`equipes.json`	JSON	Liste des équipes avec leurs statistiques
`matchs.txt`	Texte	Historique des résultats des matchs
`classement.csv`	CSV	Classement général du tournoi
`stats.txt`	Texte	Meilleure attaque et meilleure défense