Rappel sur la manipulation d'un fichier :

- Ouverture d'un fichier : nom_v=**open** (nom_fichier,mode)
  Mode :
  - '**w**' : ouverture du fichier en mode écriture si le fichier n'existe pas elle va le créer s'il existe déjà elle va écraser son contenu
  - '**a**' : ouverture de fichier en mode écriture si le fichier n'existe pas elle va le créer sinon elle va ajouter à la fin de fichier
  - '**r**' : ouverture en mode lecture si le fichier n'existe pas un message d'erreur sera lancé
- Ecriture dans un fichier : N.B le fichier doit être ouvert en mode '**a'** ou '**w'**
  - nom_v.**write** (une chaine)
- Lecture à partir d'un fichier :
  - Nom_v.**read**( ) : elle permet de retourner tous le contenu du fichier sous forme d'une chaine de caractère
  - Nom_v.**readline**( ) : elle permet de retourner une chaine de caractère qui représente une ligne de fichier (la lecture du fichier est séquentielle)
  - Nom_v.**readlines**( ) : elle permet de retourner une liste dont chaque élément de la liste représente une ligne du fichier
- La fermeture d'un fichier :
  - Nom_v.**close**( )

Problème 1(DS IPEIM):

On désire informatiser et sauvegarder les équipes participantes dans un championnat de foot ainsi que les résultats des matchs dans des fichiers.

Pour cela, on a sauvegardé le nom de chaque équipe participante dans chaque ligne du fichier **EQUIPES.txt**

Chaque équipe joue deux matchs avec chaque autre équipe adversaire : le premier match se déroule sur son terrain alors que le deuxième est joué sur le terrain de l'adversaire. Pour cela, on a sauvegardé les résultats des matchs dans un fichier **MATCH.txt** où chaque ligne comporte :

EQUIPES.txt

```
EQ1
EQ2
....
....
EQ9
EQ5
```

MATCH.txt (initial)

```
EQ1;EQ2;0:0
EQ1;EQ3;0:0
....
.....
EQ2;EQ1;0:0
EQ2;EQ3;0:0
```

Le nom de l'équipe locale ; le nom de l'équipe visiteuse ; le résultat sera une chaine de caractère dont le nombre de buts marqués par l'équipe locale : suivit par le nombre de but de l'équipe visiteuse.

Exemple d'une ligne de ce fichier :

'EQ1;EQ2;2:1' : Ce match est joué sur le terrain de 'EQ1' contre le visiteur 'EQ2' et qui a gagné 2 buts contre 1.

1) Ecrire une fonction python **INIT_MATCH** qui permet à partir du fichier **EQUIPES.txt** de créer et d'initialiser le fichier des résultats de tous les matches qui seront joués par chaque équipe enregistrées dans le fichier **MATCH.txt**. Le résultat de départ est **'0:0'**
2) Ecrire une fonction **MAJ_MATCH** qui accepte trois paramètres respectivement le nom de l'équipe locale, le nom de l'équipe visiteuse et le résultat du match (une chaine de caractères de la forme le nombre de buts marqués par l'équipe locale : suivit par le nombre de but de l'équipe

visiteuse). Elle permet de mettre à jour le fichier **MATCH.txt** sachant que l'ordre des lignes de ce fichier n'a pas d'importance.

```
EQ1;EQ2;0:0
EQ1;EQ3;0:0
....
.....
EQ2;EQ1;0:0
EQ2;EQ3;0:0
```

MATCH.txt (Avant l'appel de la fonction)

```
.........
EQ1;EQ3;0:0
....
.....
EQ1;EQ2;2:1
EQ2;EQ3;0:0
```

MATCH.txt (Après l'appel de la fonction)

3) Ecrire une fonction **RES_MATCH** qui accepte en paramètre une chaine de caractère de la forme d'une ligne du fichier **MATCH.txt** et qui permet de retourner **'1'** si l'équipe locale a gagné le match, **'2'** si l'équipe visiteuse est la gagnante sinon **'X'**.

Exemple : **RES_MATCH**('EQ2;EQ4;3:0') retourne **'1'**
RES_MATCH('EQ2;EQ4;0:0') retourne **'X'**
RES_MATCH('EQ2;EQ4;0:1') retourne **'2'**

4) Ecrire une fonction **BUT_MARQUE** qui accepte en paramètre le nom d'une équipe et qui permet de retourner le nombre des buts qu'elle a marqué à partir du fichier **MATCH.txt.**
5) Ecrire une fonction **BUT_RECU** qui accepte en paramètre le nom d'une équipe et qui permet de retourner le nombre des buts qu'elle a reçu à partir du fichier **MATCH.txt**.

Pour calculer le nombre de point obtenu dans le championnat, on accorde **3** points pour chaque match gagné, **0** pour un match perdu et **1** point pour un match nul.

6) Ecrire une fonction **NB_POINT** qui accepte en paramètre le nom d'une équipe et qui permet de retourner le nombre des points de l'équipe passé en paramètre au championnat (utiliser impérativement la fonction **RES_MATCH**)
7) Ecrire une fonction qui permet de retourner un dictionnaire dont la clé est le nom des équipes et la valeur est un tuple qui contient respectivement le nombre de points de l'équipe, buts marqués et buts reçus.

Problème 2:

Nous proposons de réaliser un programme en **Python** permettant de gérer une pharmacie dont les informations sont stockées dans des fichiers. En effet, le programme gère deux fichiers :

- **Fichier 1 : Fournisseurs.txt**

Chaque ligne de ce fichier contient les informations relatives à un fournisseur de médicaments, à savoir : son identificateur (entier), sa raison sociale (chaine), son adresse (chaine), son numéro de téléphone (chaine).

Exemple :

```
122;Parachimic;Route de Gabes, Sfax;(+216) 74 455 222
123;Adwya;Hammem sousse, Sousse;(+216) 73 152 876
```

- **Fichier 2 : Medicaments.txt**

La pharmacie dispose de plusieurs médicaments. Chaque médicament est identifié par son libellé (chaine), sa désignation (chaine), son prix de vente (réel), sa quantité disponible (entier), sa quantité minimale (entier), l'identificateur de son fournisseur (entier) et son code à barre de type EAN13 (entier).

Exemple :

```
4556;GLUCOPHAGE;Médicament pour les diabètes;30.500;53;6;123;4719512002889
4557;CLAMOXYL;Antibiotique;8.750;44;1;123;2828653219061
4558;SELEN;Complément alimentaire;23.800;3;0;122;1233131313131
.....
```

Un code à barres est la représentation graphique d'une donnée numérique ou alphanumérique. Elle apparaît sous la forme de lignes verticales, d'épaisseurs variables, généralement de couleur noire. Ces lignes forment un ensemble qui diffère suivant la symbologie utilisée. Les codes à barres sont utilisés pour leur fiabilité et la rapidité d'exécution pour la saisie des données.

En particulier, le code EAN13 (European Article Numbering à 13 chiffres) est un code à barres utilisé sur l'ensemble des produits de grande consommation. Ce code est une chaîne composée de 13 chiffres.

Un code EAN13 est formé par:

- Un identifiant du produit **q** formé par les 12 premiers chiffres à gauche.
- La clé de contrôle **cc** formée par le dernier chiffre à droite.

Pour vérifier qu'un nombre de 13 chiffres est un code EAN13 valide, on applique le principe suivant :

1. On ca1cule la somme **S** des chiffres de q en commençant par le chiffre le plus à gauche et en multipliant les chiffres de rang pair par **3**. Le rang du premier chiffre le plus à gauche est **1**, celui du deuxième chiffre le plus à gauche est **2**, etc.
2. On calcule le reste **r** de la division euclidienne de **S** par **10**.
3. On calcule **p** qui est égal à (**10 - r**).
4. Si **p** est égal à **cc** alors le code est valide.
5. Exemple : Soit le code EAN13 : **4719512002889**
   - **cc** =9
   - **q**= 4 7 1 9 5 1 2 0 0 2 8 8
   - **S** = 4+(7*3)+1+(9*3)+5+(1*3)+2+(0*3)+0+(2*3)+8+(8*3)=101
   - Le reste de la division euclidienne de S par 10 donne **r** =1
   - **p** =10- r =9
   - On remarque que **p=cc** donc le nombre 4719512002889 est un code EAN13.

NB : La gestion des exceptions seront pris en compte.

TRAVAIL A FAIRE :

Q1. Écrire une fonction **afficherTousFournisseurs** qui permet d'afficher les détails de tous les fournisseurs de cette pharmacie.

Q2. Écrire une fonction **ajouterFournisseur** qui saisit les informations d'un fournisseur à partir du clavier et les ajoute à la fin du fichier **Fournisseurs.txt**, en respectant le format donné.

Q3. Écrire une fonction **rechercherMedicament** qui permet d'afficher la liste des médicaments relatifs à un fournisseur dont la raison sociale est donnée en paramètre. Un message d'erreur sera affiché en cas d'inexistence.

Exemple : **rechercherMedicament** ('Parachimic ') Affiche tous les médicaments fournis par Parachimic

Q4. Écrire une fonction **verifierCodeEAN13** qui vérifie la validité d'un code EAN13 qui sera passé en paramètre (cette fonction retourne un booléen).

Q5. Écrire une fonction **verifierCodeEAN13MEDE** qui vérifie la validité des codes EAN13 de tous les médicaments enregistrés dans le fichier **Medicaments.txt** et qui construit un troisième fichier texte «**logcat.txt**» contenant les codes EAN13 erronés.

Q6. Écrire une fonction **menu** qui affiche le menu. Selon le choix saisi par l'utilisateur, il exécute la fonction correspondante. Le traitement se répète jusqu'à quitter l'application.

Exemple de Menu :

```
********** Menu **********
1 : Ajouter un fournisseur dans le fichier 'fourisseurs.txt'
2 : Rechercher les informations d'un fournisseur
3 : Rechercher les médicaments fournis par un fournisseur
4 : Vérifier les codes EAN13 du fichier 'medecament.txt'
5 : Quitter l'application.
**************************************************
```

Problème 3 :

Le but de cet exercice est de gérer des SMS.
Un _chier de SMS est un texte de la forme suivante :

```
0654342310;31/05/2012;10h23;Salut, t'es ou ?
0022634416;30/05/2012;11h10;asma t'a laissé un message.
0022634416;01/06/2012;09h03;T'as pas la solution de l'exo 2 ?
0654394503;01/06/2012;09h40;Qu'est-ce que tu fais, y en a marre d'attendre.
0660324524;29/05/2012;06h30;Rdv dans 5 min au 0899 bvd Royal.
0899456543;28/05/2012;06h25;vous avez gagné un téléphone
0653434325;28/04/2012;03h20;Le nouveau IPHONE à 400$ chez phonehouse.
0143523523;03/06/2012;16h23;ADIDAS: PLUS QUE 200 MODELES DISPO CHEZ DECATHLON.
```

Chaque ligne contient exactement un SMS. Les numéros de téléphone ont tous 10 chiffres, les dates sont toujours formées de 10 caractères et l'heure de 5. Des SMS seront représentés comme un dictionnaire. Les clefs sont les numéros de téléphone d'origine et les valeurs des listes de messages. Chaque message est une liste de 3 éléments [date, heure, contenu]. Choisissez une représentation adéquate pour la date et l'heure. On considère qu'un SMS est publicitaire s'il contient le caractère $ On considère qu'un message est un SPAM s'il contient un numéro de téléphone commençant avec 0899 (un numéro de téléphone à 10 chiffres).

Travail demandée :

1) écrire une fonction python **VERIF_SPAM** qui accepte en paramètre une chaine qui représentera un SMS et qui permet de vérifier est ce que c'est un SMS SPAM ou non

Exemple :**VERIF_SPAM**('0899456543;28/05/2012;06h25;vous avez gagné un téléphone') retourne **True**

- **VERIF_SPAM**(0022634416;30/05/2012;11h10;asma t'a laissé un message ') retourne **False**

2) écrire une fonction python **VERIF_PUB** qui accepte en paramètre une chaine qui représentera un SMS et de vérifier est ce que c'est un SMS publicitaire ou non

3) écrire une fonction python **NBR_SPAM** qui accepte en paramètre le nom d'un fichier texte qui contient des SMS et qui permet de retourner le nombre des SMS SPAM

4) Ecrire une fonction python **NBR_PUB** qui accepte en paramètre le nom de fichier et qui permet de retourner le nombre des SMS publicitaire

5) Ecrire une fonction python **NBR_SMS** qui accepte en paramètre le nom d'un fichier SMS et un numéro de téléphone composer de 10 chiffre et retourner le nombre des SMS venu de ce numéro

Exemple : **NBR_SMS**('SMS.txt','0022634416') retourne 2

6) Ecrire une fonction python qui permet de retourner un dictionnaire dont clé le type de l'SMS si il est SPAM ou PUBLICitaire ou le numéro de l'expéditeur et la valeur sera le nombre de sms

7) Ecrire une fonction python qui accepte en paramètre le nom de fichier et qui permet de supprimer les SMS SPAM et les SMS publicitaire