

Examen n°1

Matière : INFORMATIQUE

Classe : 2^{ème} Année BG Durée : 2h

Problème 1 : (10 points)

Soit une base de données, nommée *restaurants_reviews.db*, conçue pour analyser les avis des clients sur les plats servis dans différents restaurants. Cette base permet de suivre en ligne les clients, les plats proposés, les avis déposés, ainsi que les restaurants eux-mêmes.

Le schéma relationnel de cette base est décrit ci-dessous.

- Client(idClient, nomClient, email)

La table **Client** enregistre les informations concernant un client :

- idClient : identifie un client de type entier, clé primaire.
- nomClient : le nom du client de type chaîne de caractères.
- email : l'e-mail du client de type chaîne de caractères.

- Restaurant(idRest, nomRest, adresse)

La table **Restaurant** enregistre les informations sur les restaurants :

- idRest : identifie un restaurant de type entier, clé primaire.
- nomRest : le nom du restaurant de type chaîne de caractères.
- adresse : l'adresse du restaurant de type chaîne de caractères.

- Plat(idPlat, nomPlat, prix, idRest#)

La table **Plat** enregistre les informations concernant les plats du restaurant :

- idPlat : identifie un plat de type entier, clé primaire.
- nomPlat : le nom du plat de type chaîne de caractères.
- prix : le prix du plat de type entier supérieur à 0.
- idRest : identifie le restaurant proposant ce plat, de type entier, clé étrangère vers la table **Restaurant**.

- Avis(idAvis, idClient#, idPlat#, note, commentaire)

La table **Avis** enregistre les avis des clients :

- idAvis : identifie l'avis de type entier, clé primaire.
- idClient : identifie le client donnant l'avis de type entier, clé étrangère vers la table **Client**.
- idPlat : identifie le plat évalué de type entier, clé étrangère vers la table **Plat**.

- note : la note donnée au plat, entier de 1 à 5.
- commentaire : commentaire laissé par le client de type chaîne de caractères de taille maximale 50 caractères.

Algèbre Relationnelle : (3,5 points)

Exprimer en algèbre relationnelle les requêtes permettant de :

1. Donner les noms et les emails des clients ayant donné au moins un avis.
2. Donner les noms et les adresses des restaurants dont au moins un plat a obtenu une note de 5.
3. Afficher les noms et les adresses des restaurants qui n'ont reçu aucun avis.

SQL : (5 points)

Exprimer en SQL les requêtes suivantes permettant de :

1. Créer la table **Plat** en respectant les contraintes d'intégrités spécifiées.

Dans la suite, on suppose que les tables de la base de données sont toutes créées et remplies en respectant toutes les contraintes.

2. Donner les noms et les adresses des restaurants fournisseurs de plat 'Ojja aux fruits de mer' avec prix du plat.
3. Lister les noms des 10 plats les moins chers avec les noms des restaurants fournisseurs dont l'adresse du restaurant contient le mot « Sfax ».
4. Lister les plats qui ont reçu une moyenne de notes strictement supérieure à 4. Ici, on souhaite afficher le nom du restaurant fournisseur, son adresse, le nom du plat, et le prix.
5. Donner le nom et l'email du client qui a laissé le plus d'avis.

SOLite : (1.5 points)

On note *cur* le curseur d'exécution de requêtes avec le module *sqlite3* de Python. Ecrire une fonction **meilleurRestaurant** qui prend en paramètre *cur*, le nom d'un plat **nom** et affiche les noms et les adresses des restaurants ayant obtenu une note ≥ 4 pour ce plat.

Problème 2 : (10 points)

Dans ce problème, on s'intéresse à implémenter quelques fonctions qui aident à analyser et interpréter des courbes sauvegardées sous forme d'images **png**, dont les valeurs des abscisses et des ordonnées sont perdues.

Rappelons qu'une image est une collection de pixels organisés sous forme matricielle où chaque pixel est défini par :

- Ses coordonnées (x,y) avec x l'indice de ligne et y l'indice de colonne
- Sa couleur codée en **R** (rouge), **V** (vert), **B** (bleu). La valeur de RVB varie entre 0 et 255. Si $R=V=B=0$ alors le pixel est noir. Si $R=V=B=255$ alors le pixel est blanc.

On suppose que pour les images qu'on va traiter :

- La courbe est tracée en bleu pur. Les pixels représentant la courbe ont la valeur [0,0,255].

- Le reste de l'image est formé par des pixels blancs [255,255,255] ou bien gris [x,x,x] avec x un entier compris entre 0 et 255.
- Toutes les images sont similaires au modèle représenté par la figure 1.

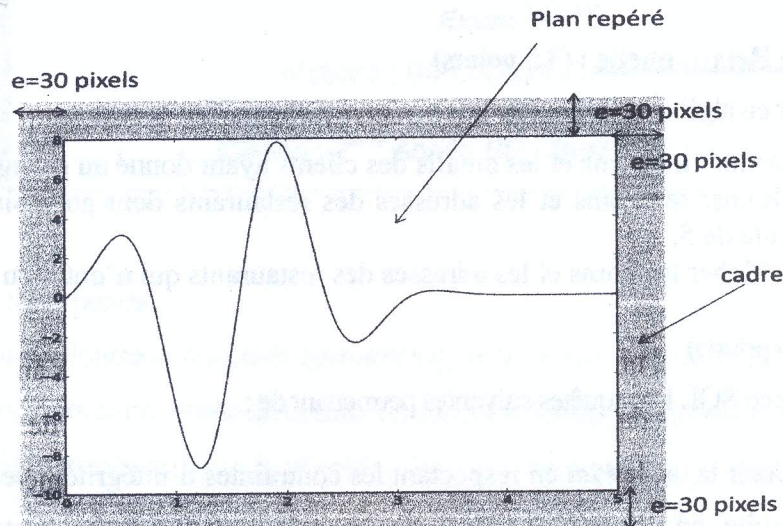


Figure 1

Hypothèse de travail

- Les réponses seront écrites en langage Python.
- Les matrices des images à traiter sont de type *ndarray* (tableau) du module *numpy*.
- $M[i,j]$ est un tableau de trois entiers qui codent la couleur du pixel.
- Toutes les figures ont les mêmes dimensions (hauteur et largeur).
- Toutes les figures ont les mêmes dimensions des plans repérés.

Indication

Pour vérifier qu'un pixel de coordonnées (i,j) est bleu pur, on peut utiliser la condition :
`list(M[i,j])==[0,0,255]`

Travail demandé

1. Ecrire une fonction, nommée *sansCadre*, qui prend en entrée une matrice *img* représentant une image et un entier *e* représentant l'épaisseur du cadre en pixel et retourne une image sans cadre.

Exemple : Pour l'exemple de la figure 1, cette fonction peut supprimer le cadre d'épaisseur *e=30 pixels* et on obtient alors le plan repéré avec la courbe sans les deux axes comme le montre la figure 2 (A).

2. Ecrire une fonction, nommée *inverser*, qui prend en entrée une matrice *img* représentant une image et renvoie sa transformée par la symétrie d'axe horizontal.

Exemple : Pour l'exemple de la figure 2 (A), cette fonction retourne la matrice représentant la figure 2 (B).

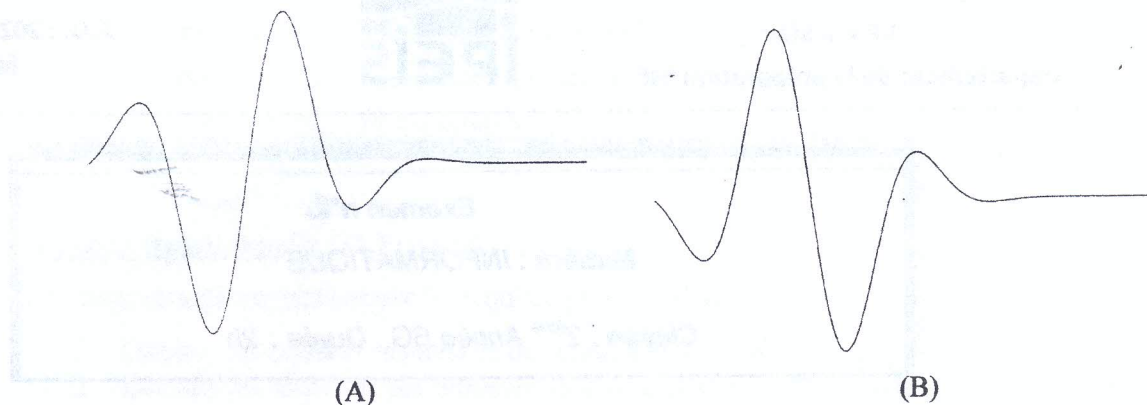


Figure 2

3. Ecrire une fonction, nommée **colorer**, qui prend en entrée une matrice *img* représentant une image de courbe en bleu pur (sans cadre) et renvoie une matrice représentant la même courbe en vert pur.
4. Ecrire une fonction, nommée **combiner**, qui prend en entrée deux matrices *img1* et *img2* représentant deux images de courbes et renvoie une matrice représentant une image des deux courbes combinées en mettant une ligne horizontale noire d'épaisseur 3 pixels entre les deux courbes (voir figure 3 (A)).

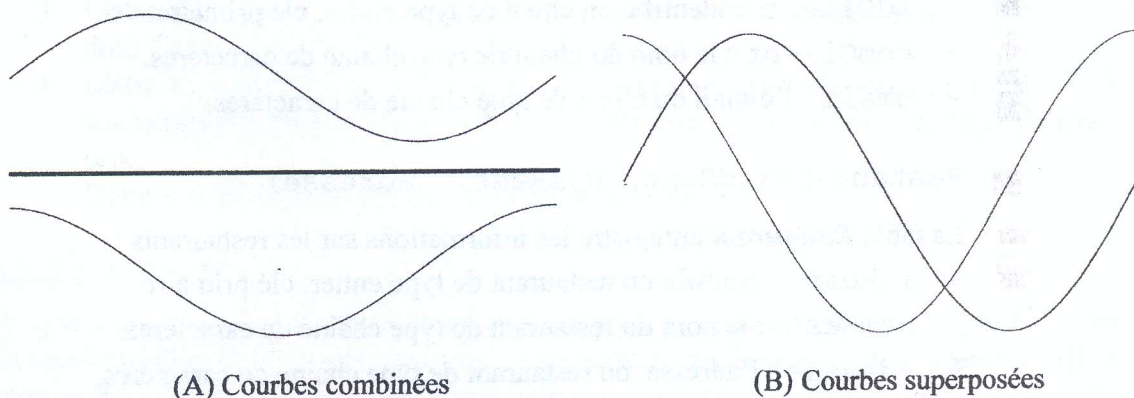


Figure 3

5. Ecrire une fonction, nommée **superposer**, qui prend en entrée deux matrices *img1* et *img2* représentant deux images de courbes et renvoie une matrice représentant une image des deux courbes superposées (voir figure 3 (B)).
6. Ecrire une fonction, nommée **echantillon**, qui prend en paramètre :
 - Une matrice *img* représentant une image de courbe
 - un entier *d* représentant l'indice du début de l'échantillon
 - un entier *f* représentant l'indice de fin de l'échantillon

Cette fonction permet d'extraire un échantillon d'une courbe. Elle retourne alors une matrice ayant le même nombre de lignes que *img* et *f-d+1* colonnes.

Bonne chance