Informatique tronc commun : Projet

Mai - Juin 2017

1 Organisation

1.1 Déroulement

Ce projet se déroule de la semaine du 15 mai à la semaine du 19 juin (incluses). Vous trouverez sur les sites des deux classes les calendriers actualisés des séances de TP.

Nous vous demandons d'assister à trois séances, au minimum. Vous êtes libres de choisir la date des séances de TP dans la limite des places disponibles (les étudiants devant avoir TP ce jours là étant prioritaires).

Vous devrez rendre votre projet au plus tard le jeudi 22 juin à minuit. Ces projets seront évalués. Les conseils de classe n'ayant lieu que quelques jours après cette date du 22 juin, n'hésitez pas à rendre votre projet plus tôt si vous avez fini en avance, cela facilitera le travail de correction de vos enseignants.

Attention :

— Les projets incomplets ou en retard seront notés 0 et/ou signalés sur le bulletins.

— En pratique, un projet informatique est source d'innombrables problèmes. Ces problèmes doivent être anticipés, ce qui est difficile car ils sont imprévisibles. Il faut donc essayer de finir en avance sur la date prévue pour avoir une marge de sécurité. Si aucun problème ne se pose, vous aurez alors la possibilité de peaufiner votre projet. **En aucun cas, ces problèmes ne sont une excuse valable pour rendre en retard**. Par exemple, si votre ordinateur plante la veille du jour du rendu et que vous vous apercevez alors que la clé USB sur laquelle vous faisiez vos sauvegardes est passée à la machine à laver, cela n'est pas une excuse valable.

1.2 Équipes

Le projet se fait par binômes, les binômes devant être dans le même groupe de TP. Si le nombre de membres de votre groupe de TP est impair (et seulement dans ce cas), votre enseignant peut autoriser un unique monôme ou un unique trinôme (consultez-le immédiatement pour savoir si c'est votre cas).

1.3 Ce qui est attendu

1.3.1 Première séance de TP

Choisir un sujet et s'entraîner à manipuler des bases de données SQL avec `sqlite3` et `Python` en utilisant la base de données de films présentée en cours.

1.3.2 Deuxième séance

Lors de la deuxième séance de TP, vous devez déjà avoir largement commencé votre projet. Vous devez avoir sur vous un cahier des charges dactylographié indiquant :

1. Vos noms.
2. Le sujet que vous avez choisi.
3. Ce que vous comptez faire sur le sujet (à quoi votre programme ressemblera pour l'utilisateur final).

Vous pourrez alors poser des questions à votre enseignant. Si celui-ci valide votre cahier des charges, vous lui rendrez. Il pourra également, s'il l'estime préférable, vous demander d'y apporter des modifications et vous dira dans ce cas combien de temps il vous laisse pour faire ces modifications et lui rendre la nouvelle version.

1.3.3 Troisième séance

C'est le moment de résoudre les derniers petits problèmes avec l'aide de votre enseignant et de voir si le rapport que vous avez rédigé convient ou ce qu'il faut adapter à ce rapport.

1.3.4 Après la troisième séance

Au moment de rendre votre TP, vous devez rendre à votre enseignant :

1. Un compte-rendu dactylographié, au format PDF.
2. Votre programme python.
3. Votre base de données.

Le nom de votre fichier PDF sera impérativement de la forme `dupont-durand-projet.pdf`, celui de votre fichier python `dupont-durand-projet.py` et celui de votre base de données `dupont-durand-projet.db`, où `dupont` et `durand` sont à remplacer par vos noms (en minuscules et sans caractères accentués).

2 Sujet

2.1 Contraintes

Vous pouvez choisir le sujet que vous voulez sous réserve du respect absolu des contraintes suivantes (en cas de doute, demandez à votre enseignant) :

1. Votre programme doit être écrit en python version 3. **Pas python2**.
2. Pour lancer votre programme, il doit être suffisant d'ouvrir un terminal, de se placer dans le répertoire où se situe votre fichier `dupont-durand-projet.py` et de taper `python3 dupont-durand-projet.py`.

3. Votre programme doit manipuler une base de données comportant plusieurs tables et certaines des requêtes que vous ferez sur cette base de données doivent être non triviales. On considérera qu'une requête qui manipule des données venant d'au moins deux tables différentes est non triviale. Les tables de votre base doivent être raisonnablement remplies (on ne vous demande pas des milliers de données mais si vos tables ne contiennent que 5 lignes chacune, il y a sans doute un problème).
4. Votre programme doit être un programme utilisable par quelqu'un qui ne connaît rien à python et doit lui apporter (un petit) quelque chose.

2.2 Exemples de sujets

Voici quelques exemples de sujets. Vous pouvez prendre un sujet totalement différent, vous pouvez également prendre un des sujets suggérés ci-dessous et l'adapter à votre goût.

2.2.1 Manipulation d'une base de données cinématographiques

Grâce à `sqlite3`, vous pouvez créer un fichier de base de données `cinema.db` comportant les tables présentées en cours, puis vous pourrez importer les données fournies dans les fichiers `.csv` du TP précédent. Puis vous écrirez en Python un programme qui permettra à un utilisateur de consulter voire de modifier cette base de données.

2.2.2 Gestion de recettes de cuisine

Vous créerez une base comportant des recettes de cuisine. Pour chaque recette, il pourra être intéressant de fournir la liste des ingrédients possible (et des quantités). On pourrait vouloir chercher des recettes comportant certains ingrédients ou au contraire ne comportant pas certains ingrédients. On peut aussi imaginer d'associer à chaque ingrédient un prix et utiliser les capacités de SQL pour calculer le prix d'une recette en fonction des ingrédients et des quantités, etc.

2.2.3 Micro-APB

Créer une base représentant une liste de candidats pour une formation. Aux différents candidats, on veut pouvoir associer son lycée, les matières qu'il a suivi et les moyennes qu'il a eues dans ces matières en terminale (premier et second trimestre), ainsi que son classement dans cette matière et l'effectif de son groupe-classe pour cette matière. On veut ensuite pouvoir trier des candidats par ordre de moyenne, mettre une priorité (donc un classement) pour les places d'internat en fonction de sa moyenne générale et de la distance de sa ville d'origine (qu'on estimera être celle de son lycée) à la ville où a lieu la formation demandée. Pour calculer cette distance, on pourra gérer une table de villes avec leurs coordonnées géographiques et se contenter de calculer la distance à vol d'oiseau.

2.2.4 Gestion de chambres d'hôtel

On veut gérer une base de données d'hôtels. Chaque hôtel est dans une ville, possède des chambres de différentes catégories, à différents prix. On voudrait par exemple proposer

de chercher une chambre la moins chère possible de telle catégorie dans telle ville ou une chambre de la catégorige la plus élevée possible pour tel budget et, bien sûr, la réserver.

3 Quelques informations utiles

3.1 Sqlite3

Dans un terminal, se placer dans le répertoire où l'on veut travailler (rappel : les commandes `cd`, `pwd` et `mkdir` sont utiles). Le frontal en ligne de commande pour SQLite s'appelle `sqlite3` (lancer `man sqlite3` pour voir le manuel).

On peut lancer `sqlite3 dupont-durand-projet.db` pour ouvrir la base de données `dupont-durand-projet.db`. Si elle n'existe pas, elle est créée (pour peu qu'on y crée une table).

3.2 Utilisation d'une base depuis Python

Comme vu en cours, le module python à utiliser se nomme `sqlite3`, il est dans la bibliothèque standard de Python (et sa documentation avec celle de la bibliothèque de Python).

3.3 Interagir avec un utilisateur

Vous connaissez la commande `print` qui permet d'afficher un texte.

La fonction `input()` attend la saisie d'une ligne par l'utilisateur (terminée par un retour chariot) et retourne pour valeur la chaîne de caractères rentrée par l'utilisateur.

Exemple :

```
def carre(x):
  """Retourne le carré de son argument"""
  return x**2

def calcule_carre():
  """Demande un nombre à l'utilisateur,\
    calcule son carré et l'affiche.
  Cette fonction ne gère pas très bien les problèmes,\
    elle déclenche une erreur si l'utilisateur ne\
    rentre pas un nombre."""
  x = float(raw_input())
  print(carre(x))

calcule_carre()
```