# Programmation C avancée : Bibliothèques statiques et dynamiques

Nous allons utiliser le code généré précédemment pour illustrer le principe des bibliothèques statiques et dynamiques.

Le but est de rajouter deux fichiers spécifiques `student_api.c` et `student_api.h` qui vont contenir les fonctions précises pour faire l'interface entre l'utilisateur et votre code. Cette interface doit être respectée à la lettre de manière à pouvoir l'utiliser de manière transparente d'un code à l'autre.

## NOMMAGE DE LA BIBLIOTHEQUE

La bibliothèque portera le nom de :

— lib**student_s**.a pour une bibliothèque statique.

— lib**student_d**.so pour une bibliothèque dynamique.

## FONCTIONS D'INTERFACE

1. *CLASS_DATA** **API_load_students**(*char** filePath) ;
   Cette fonction va prendre en entrée le chemin du fichier texte contenant les données des étudiants et va charger toutes les données en RAM. Ensuite elle va retourner un pointeur de type *CLASS_DATA*. Ce type de structure sera redéfini par rapport à votre propre code.
   Si une erreur survient, la valeur *NULL* sera retournée.

2. *int* **API_save_to_binary_file**(*CLASS_DATA** pClass, char* filePath) ;
   Cette fonction va sauvegarder l'ensemble des données en mémoire dans un fichier texte dont le nom est passé en paramètre.
   La valeur de retour sera un booléen pour indiquer si la fonction a été exécutée avec succès ou non.

3. *CLASS_DATA** **API_restore_from_binary_file**(char* filePath) ;
   Cette fonction va restaurer en mémoire l'ensemble des données depuis un fichier binaire dont le nom est passé en paramètre. La valeur de retour sera un pointeur vers la structure *CLASS_DATA* comme avec la fonction **API_load_students()**, ou *NULL* si une erreur est rencontrée.

4. *void* **API_display**(*CLASS_DATA** pClass) ;
   Cette fonction va afficher l'ensemble des données en mémoire. Le format est laissé libre du moment qu'il respecte les conditions minimales suivantes :
   — Afficher la version de la bibliothèque avec la référence de l'auteur (à minima votre nom pour savoir quelle bibliothèque est utilisée)
   — Afficher le nombre de matières et leurs noms avec les coefficients respectifs
   — Afficher le nombre d'étudiants ainsi que leurs noms suivis deleurs moyennes (chaque matière + moyenne générale)

5. *void* **API_unload**(*CLASS_DATA** pClass) ;
   Cette fonction va simplement effectuer la libération de toute la mémoire déjà allouée grâce à **API_load_students()** ou **API_restore_from_binary_file()**.

6. *char*** **API_get_best_students**(*CLASS_DATA** pClass) ;
   Cette fonction va retourner un tableau de *SIZE_TOP1* chaines de caractères (cette taille doit définie dans votre fichier d'entête). Fixer cette taille à 10 dans votre code.
   L'ensemble des données retournées sont allouées dynamiquement à chaque appel. Il est de la responsabilité du code appelant de libérer la mémoire allouée (chaque chaine de caractères ainsi que le tableau englobant les chaines).
   Le format de chaque chaine de caractères est laissé libre.

7. *char*** **API_get_best_students_in_course**(*CLASS_DATA** pClass, **char*** course) ;
   Cette fonction va retourner un tableau de *SIZE_TOP2* chaines de caractères (cette taille doit définie dans votre fichier d'entête). Fixer cette taille à 3 dans votre code.
   L'ensemble des données retournées sont allouées dynamiquement à chaque appel. Il est de la responsabilité du code appelant de libérer la mémoire allouée (chaque chaine de caractères ainsi que le tableau englobant les chaines).
   Le format de chaque chaine de caractères est laissé libre.