ECM - S7 - PROJECT C++ - 2021

Gestion des bibliothèques

Réalisé par :

- Yassine AMCHAROD
- Hind CHOUKRI

08 Janvier 2021

TABLES DE MATIERES

I. Introduction

1.1 Contexte

Ce projet s'inscrit dans le cadre de l'UE « Programmation orientée objet :C++ ».Le but de ce projet est de fournir un programme qui permette de gérer un réseau de bibliothèques en utilisant les notions acquises pendant cet UE.

1.2 Présentation du projet

Le projet consiste à représenter les différentes bibliothèques et aussi leur contenu, leurs adhérents, les emprunts, les échanges entre bibliothèques, les achats, les pertes et la mise au pilon de certains ouvrages.

1.3 Travail réalisé

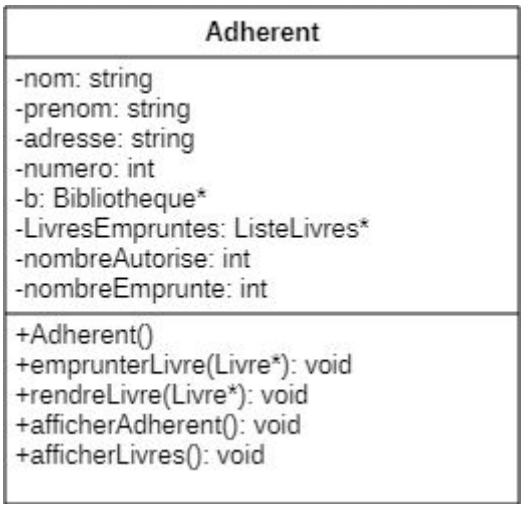
Dans un premier temps, on présente les classes utilisées ,leurs diagrammes UML et un diagramme UML représentant la liaison entre toutes ces classes. Ensuite, on va présenter les différentes méthodes,et fonctions utilisées pour réaliser le travail demandé, tout en décrivant leurs rôles et leur fonctionnement global.

II. Modélisation du projet

1.1 Classes & diagrammes UML

- **Classe Adherent :** la classe qui représente un adhérent d'une bibliothèque ,tel que chaque adhérent est représenté par son nom, son prénom, son adresse, son numéro d'adhérent, la bibliothèque à laquelle il est inscrit, la liste des livres empruntés et le nombre de livres autorisé à l'emprunt.

  ➔ **Diagramme UML :**

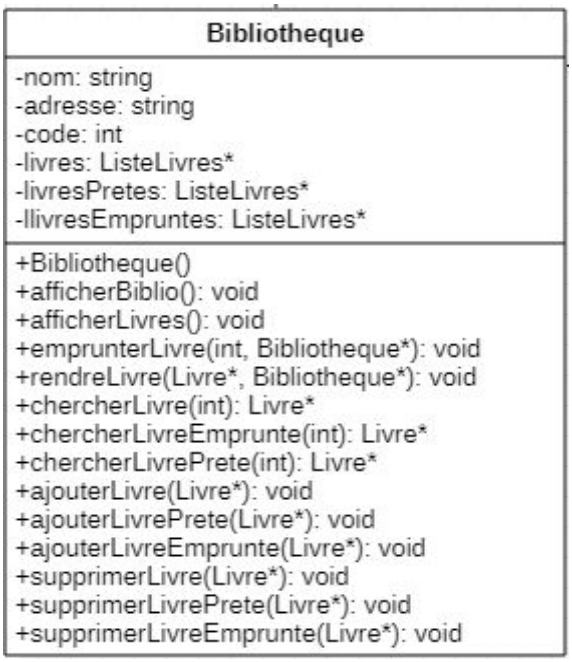


  ➔ **Code :**

```
class Adherent {
private:
    string nom;
    string prenom;
    string adresse;
    int numero;
    Bibliotheque*  b;
    ListeLivres* livresEmpruntes;
    int nombreAutorise;
    int nombreEmprunte;
public:
    Adherent(string n, string p, string a, int num, Bibliotheque* bib, int nbrMax);
    void emprunterLivre(Livre* l);
    void rendreLivre(Livre* l);
    void afficherAdherent();
    void afficherLivres();
    virtual ~Adherent();
```

- **Classe Bibliotheque :** la classe qui représente une bibliothèque ,tel que chaque bibliothèque est représentée par son nom, son adresse, son code et la liste de ses livres.

➔ Diagramme UML :

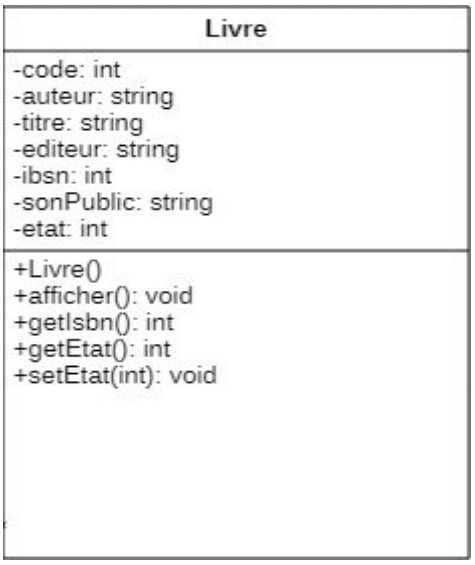


➔ Code :

```
class Bibliotheque {
private:
    string nom;
    string adresse;
    int code;
    ListeLivres* livres;
    ListeLivres* livresPretes;
    ListeLivres* livresEmpruntes;

public:
    Bibliotheque(string n, string a, int c);
    void afficherBiblio();
    void afficherLivres();
    void emprunterLivre(int isbn, Bibliotheque* b);
    void rendreLivre(Livre* l, Bibliotheque* b);
    Livre* chercherLivre(int isbn);
    Livre* chercherLivreEmprunte(int isbn);
    Livre* chercherLivrePrete(int isbn);
    void ajouterLivre(Livre* l);
    void ajouterLivrePrete(Livre* l);
    void ajouterLivreEmprunte(Livre* l);
    void supprimerLivre(Livre* l);
    void supprimerLivrePrete(Livre* l);
    void supprimerLivreEmprunte(Livre* l);

    virtual ~Bibliotheque();
};
```

- **Classe Livre :** la classe qui représente un livre d'une bibliothèque.Chaque livre est représenté par son code (nécessaire si plusieurs exemplaires),son auteur, son titre, son éditeur, son numéro ISBN, son public : adulte, ados, jeunesses ou tout public, son état (libre ou emprunté ou prêté).

  ➔ **Diagramme UML :**

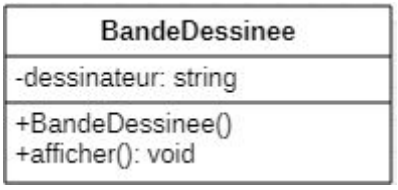


➔ **Code :**

```
class Livre {
private:
    int code;
    string auteur;
    string titre;
    string editeur;
    int isbn;
    string sonPublic;
    int etat; // etat = 0 : libre, etat = 1: emprunté, etat = 2 : preté

public:
    Livre(int c, string a, string t, string e, int i, string pub);
    virtual void afficher();
    int getIsbn();
    int getEtat();
    void setEtat(int newEtat);
    virtual ~Livre();
};
```

- **Classe BandeDessinee :** est une classe représentant une bande dessinée qui est un livre avec un dessinateur. Cette classe hérite de la classe Livre.

  ➔ **Diagramme UML :**

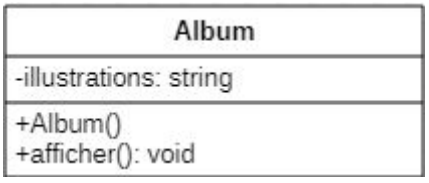


  ➔ **Code :**

```
class BandeDessinee: public Livre {
private:
    string dessinateur;
public:
    BandeDessinee(int c ,string a , string t , string e , int i , string p ,string d);
    virtual void afficher();
    virtual ~BandeDessinee();
};
```

- **Classe Album :** est une classe représentant un album en héritant de la classe Livre ,en ajoutant un attribut illustrations pour spécifier si elles s'agissent des photos ou des dessins ou les deux.

  ➔ **Diagramme UML :**

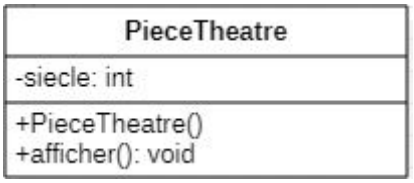


  ➔ **Code :**

```
using namespace std;
class Album: public Livre {
private:
    string illustrations;
public:
    Album(int c ,string a , string t , string e , int i , string p, string illus);
    void afficher();
    virtual ~Album();
};
```

- **Classe PieceTheatre :** est une classe représentant une pièce de théâtre qui est un livre caractérisé par l'attribut siècle.Cette classe hérite de la classe Livre.

  ➔ **Diagramme UML :**

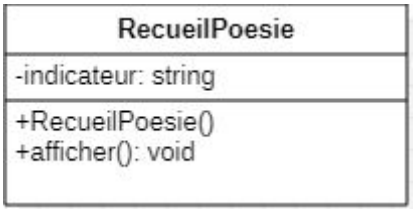


  ➔ **Code :**

```
class PieceTheatre: public Livre {
private:
    int siecle;
public:
    PieceTheatre(int c ,string a , string t , string e , int i , string p, int s);
    virtual void afficher();
    virtual ~PieceTheatre();
};
```

- **Classe RecueilPoesie :** la classe représentant un recueil de poésie qui est un livre avec un indicateur : vers ou prose ou bien les deux.Elle hérite de la classe Livre.

  ➔ **Diagramme UML :**

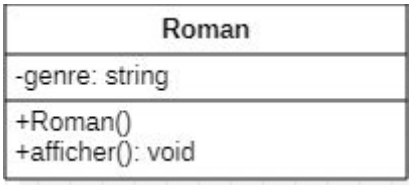


  ➔ **Code :**

```
class RecueilPoesie: public Livre {
private:
    string indicateur;
public:
    RecueilPoesie(int c ,string a , string t , string e , int i , string p, string indic);
    virtual void afficher();
    virtual ~RecueilPoesie();
};
```

- **Classe Roman :** La classe représentant un roman qui est un livre avec un genre : Littérature, Roman noir, Roman policier, Roman animalier, Roman d'amour, Roman de mœurs, Roman gothique, Roman courtois, Roman épistolaire, Roman-feuilleton, Roman graphique, Roman historique, Roman-photo, Roman picaresque, Roman-mémoires, Roman populaire, Roman d'aventures, Roman d'anticipation, Roman d'espionnage, Roman d'apprentissage, Roman de chevalerie, Roman autobiographique, Nouveau roman, Roman chevaleresque, Conte, Nouvelle etc...Cette classe hérite également de la classe Livre.

  ➔ **Diagramme UML :**



  ➔ **Code :**

```
class Roman: public Livre {
private:
    string genre;
public:
    Roman(int c ,string a , string t , string e , int i , string p, string type);
    virtual void afficher();
    virtual ~Roman();
};
```

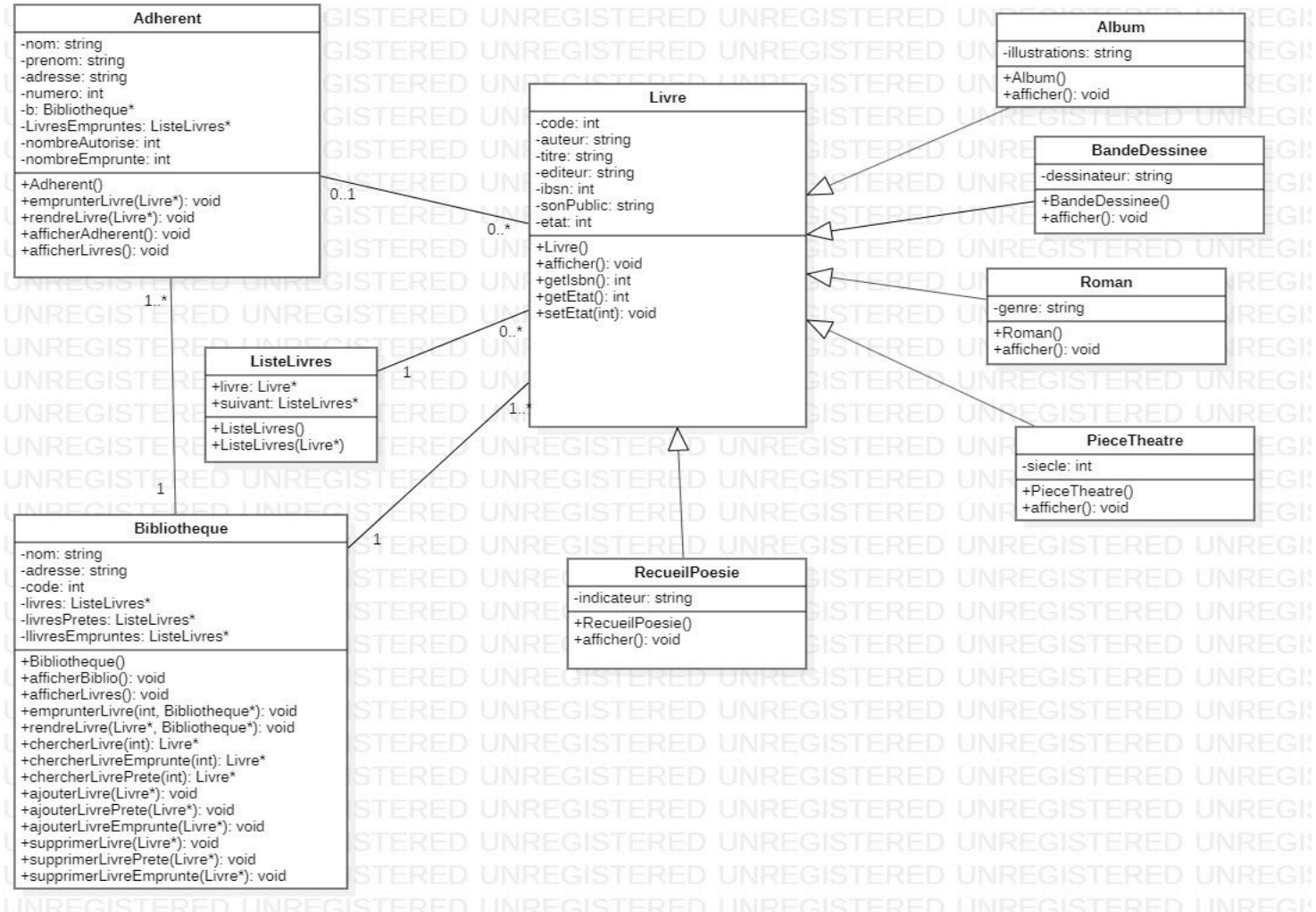
- **Classe ListeLivres :**

  ➔ **Diagramme UML :**

  ➔ **Code :**

```
class ListeLivres {
public:
    Livre* livre;
    ListeLivres* suivant;
    ListeLivres();
    ListeLivres(Livre* l);
    virtual ~ListeLivres();
};
```

1.2 Diagramme UML du projet

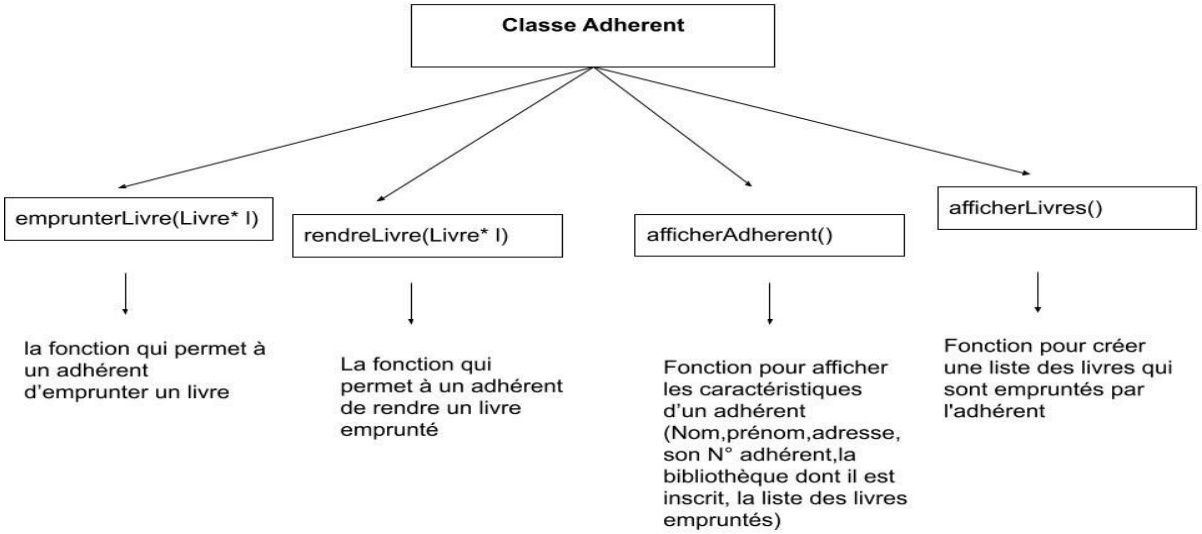


- Tels que :
  - ➔ **Relation Livre/Album, BandeDessinee, Roman, PieceTheatre, RecueilPoesie :** Les classes Album, BandeDessinee, Roman, PieceTheatre, RecueilPoesie héritent de la classe Livre.
  - ➔ **Relation Adherent/Livre :** Un adhérent peut emprunter ou rendre un ou plusieurs livres(max : nbreEmprunte).
  - ➔ **Relation Adherent/Bibliotheque :** Un adhérent (à plusieurs)sont inscrits dans une bibliothèque.
  - ➔ **Relation Bibliotheque/Livre :** Chaque bibliothèque contient plusieurs livres.
  - ➔ **Relation ListeLivres/Livre :** Chaque ListeLivres est composé de plusieurs livres.

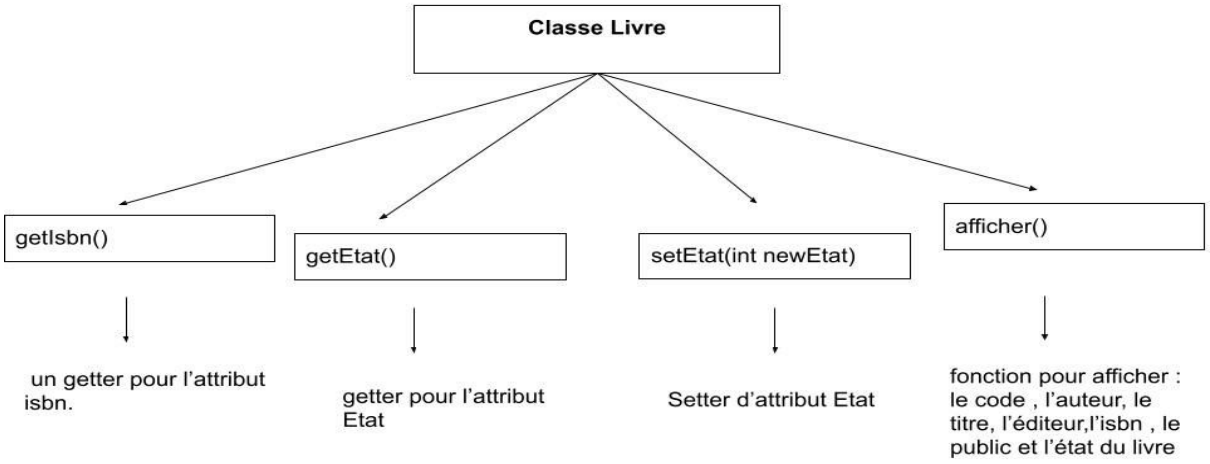
III. Réalisation du projet

1.1 Méthodes des classes

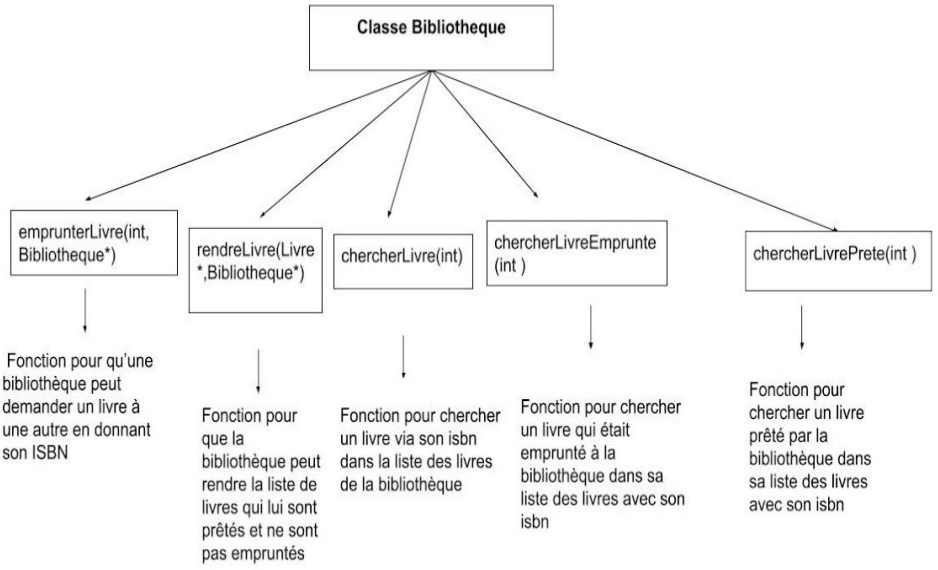
- **Classe Adherent :**



- **Classe Livre :**



- **Classe Bibliotheque :**



En exécutant le projet on obtient dans un premier temps le menu suivant:

```
###### Projet Bibliotheque (Amcharod et Choukri) ######

===== Menu =====
1) Programmation sans interface utilisateur
2) Programmation avec interface
3) Programmation avec stockage dans des fichiers
entrez Votre choix (1 ou 2 ou 3 ) :
```

1.2 Programmation sans interface utilisateur

Pour cette première version il n' y aura pas une interaction de l'utilisateur,car tout est réalisé avec aucune saisie externe en entrant tous les objets au début du programme. Si on saisit le premier choix on obtient les pages ci-dessous:

```
entrez Votre choix (1 ou 2 ou 3 ) :1
===== Tests =====

>>>> test Affichage d'une bibliotheque et de ses livres :

nom BibliothPque : saint charles
Adresse : marseille centre ville
Code  : 1
>>livres :
### Roman :
code : 3
auteur : taha
titre : html
editeur : simi
isbn : 33
public : adulte
Útat : 0
genre : drama
###Bande DessinÚe :
code : 2
auteur : yassine
titre : jee
editeur : baka
isbn : 22
public : adulte
Útat : 0
dessinateur : jack
code : 1
auteur : romain
titre : java
editeur : nada
isbn : 11
public : adulte
Útat : 0
>>>> test des opÚrations par les adherents :
```

➔ **Test des opérations des adhérents donne:**

```
Le livre n'est pas disponible dans votre bibliothPque
Vous avez dÚpacÚ le nombre de Livres autorisÚs ( 2 ).
Le livre n'est pas disponible pour l'instant
###Bande DessinÚe :
code : 2
auteur : yassine
titre : jee
editeur : baka
isbn : 22
public : adulte
Útat : 1
dessinateur : jack
code : 1
auteur : romain
titre : java
editeur : nada
isbn : 11
public : adulte
Útat : 1
```

➔ **Test de suppression d'un livre :**

```
>>>> test supprimer livre :

###Bande DessinÚe :
code : 2
auteur : yassine
titre : jee
editeur : baka
isbn : 22
public : adulte
Útat : 1
dessinateur : jack
code : 1
auteur : romain
titre : java
editeur : nada
isbn : 11
public : adulte
Útat : 1
```

➔ **Test pour les différents emprunts effectués par genre de livre :**

```
>>>> test des empruntes rÚalisÚs entre les biblios :

Le livre est deja empruntÚ par adherent de votre bibliotheque
### Album :
code : 4
auteur : arthur
titre : css
editeur : doha
isbn : 44
public : adulte
Útat : 0
illustrations :photo
###Recueil de PoÚsie :
code : 6
auteur : ali
titre : php
editeur : tawfik
isbn : 66
public : adulte
Útat : 0
indicateur : vers
### PiÚce de thÚÔtre
code : 5
auteur : salah
titre : javascript
editeur : rajib
isbn : 55
public : adulte
Útat : 0
 siPcle : 19
########
```

1.3 Programmation avec interface

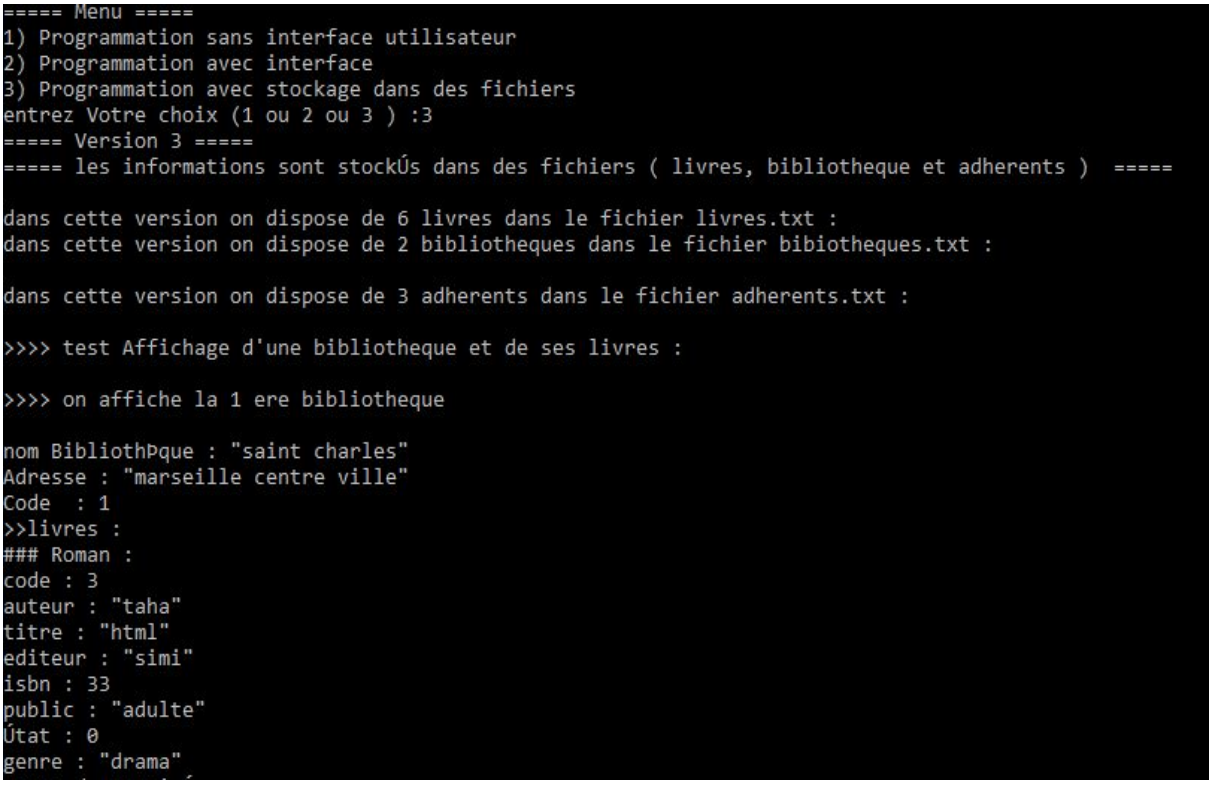
Pour cette version, l'utilisateur devrait saisir mais tout en respectant la structure précédente. En saisissant cette fois le deuxième choix on obtient les pages ci-dessous:

```
###### Projet Bibliotheque (Amcharod et Choukri) ######

===== Menu =====
1) Programmation sans interface utilisateur
2) Programmation avec interface
3) Programmation avec stockage dans des fichiers
entrez Votre choix (1 ou 2 ou 3 ) :2
===== Version 2 =====
===== Entrer les livres =====

dans cette version vous devez entrer 5 livres :
>>>Entrer le Livre numero 1
quel est Votre choix :
1) Bande Dessinee
2) Roman
3) Album
4) Piece de Theatre
5) Recueil de Poesie
```

```
dans cette version vous devez entrer 5 livres :
>>>Entrer le Livre numero 1
quel est Votre choix :
1) Bande Dessinee
2) Roman
3) Album
4) Piece de Theatre
5) Recueil de Poesie
2
Entrer code :
3
Entrer auteur :
taha
Entrer titre :
html
Entrer edition :
simi
Entrer isbn :
33
Entrer publique :
adulte
Entrer genre :
drama
```

1.4 Programmation avec stockage dans des fichiers

Cette troisième version consiste à stocker quelques livres, quelques bibliothèques et quelques adhérents dans des fichiers.On obtient les résultats suivants quand on choisit cette version :



IV. Conclusion

Ce projet fut une très bonne expérience, vu qu'il a testé nos connaissances acquises lors de l'UE "Programmation orientée Objet C++" et on a également pu améliorer toutes nos compétences en c++ et il nous a permis aussi de se familiariser plus avec .