Explications

Projet Bibliothèque

I/ Table des matières .c

8. FONCTIONS de recherche diverses

8.1 void rechercher(Lecteur **tab, int nb, Ouvrage **tabOuv, int nbOuv, ListeEmprunt l); ….1327

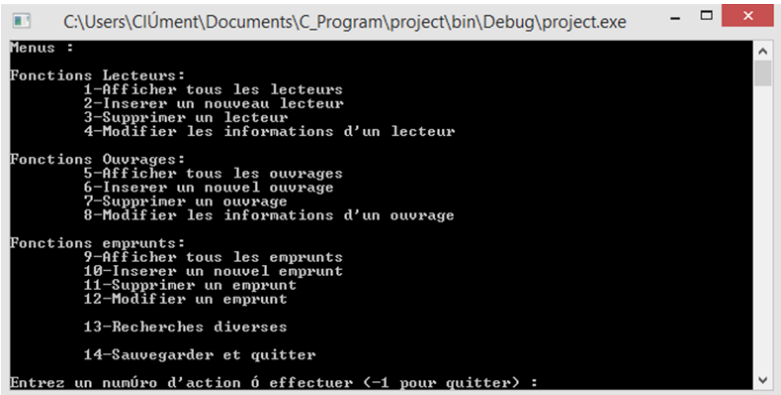
9. GLOBAL

9.1 void global(void) ……1470

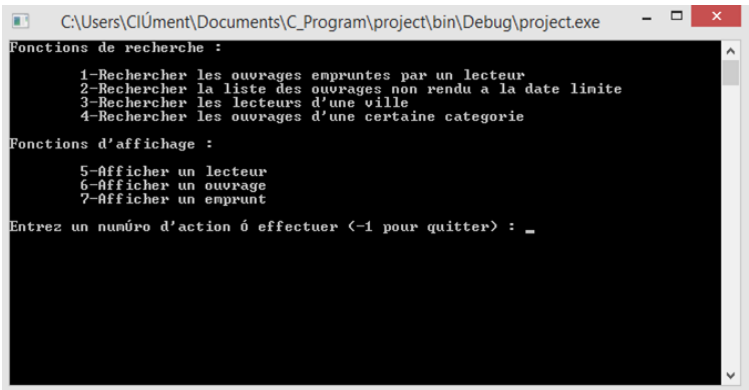
II/ Modèle entité-association

III/ Affichage

Menu :

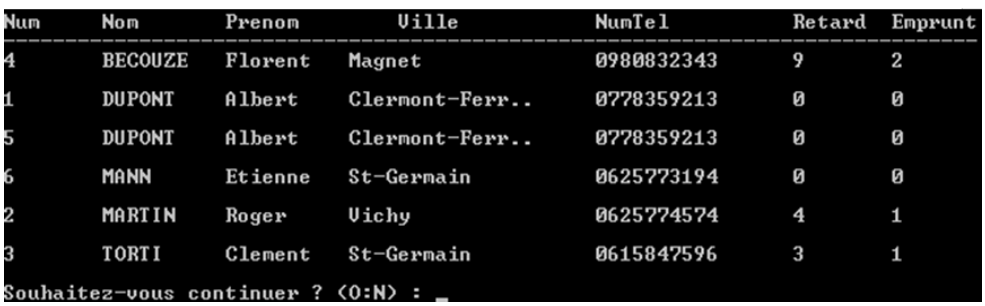


Sous-Menu :



Tableaux :

Lecteurs :



Ouvrages :

```
Cote    Titre           Categorie       Auteur          Emprunte(Oui/Non)
-------------------------------------------------------------------------
B10B3   Tintin au Tib.. BD              Herge           OUI
F20C4   Harry Potter .. Fantaisi..      JK Rowli..      OUI
S50F6   Soul Eater      Manga           Okubo           OUI
S52F1   Sciences & Vi.. Science         *****           OUI
T123B   Lapin cretin    BD              ???             NON
Souhaitez-vous continuer ? (O:N) : _
```

Emprunts :

```
Num     Cote    Date Emprunt    Date Retour
-------------------------------------------
2       F20C4   14/09/17        14/10/17
3       S52F1   02/05/17        02/06/17
4       B10B3   27/12/17        30/01/18
4       S50F6   26/03/17        26/04/17

Souhaitez-vous continuer ? (O:N) : _
```

Fichiers :

Lecteurs :

```
5
DUPONT
Albert
Clermont-Ferrand
0778359213
0
0
6
MANN
Etienne
St-Germain
0625773194
0
0
```

Ouvrages :

```
5
B10B3
Tintin au Tibet
BD
Herge
O
F20C4
Harry Potter 4
Fantaisie
JK Rowling
O
S50F6
Soul Eater
Manga
Okubo
O
```

Emprunts :

```
STXNULNULNULF20C4NULSONULNULNUL
NULNULNULDC1NULNULNULSONULNULNUL
NULNULNULDC1NULNULNULETXNULNULNULS52F1NULSTXNULNULNULENQNULNUL
NULDC1NULNULNULSTXNULNULNULACKNULNULNULDC1NULNULNULEOTNULNULNUL
B10B3NULESCNULNULNULFFNULNULNULDC1NULNULNULRSNULNULNULSOHNULNUL
NULDC2NULNULNULEOTNULNULNULS50F6NULSUBNULNULNULETXNULNULNULDC1
NULNULNULSUBNULNULNULEOTNULNULNULDC1NULNULNUL
```

IV/ Stratégie

- **Lecteurs :**
  - Les lecteurs sont enregistrés dans un fichier texte et chargés dans un tableau dynamique de pointeur vers des structures Lecteur.
  - On ne connait pas le nombre de lecteur à l'ouverture du fichier.
  - Les lecteurs sont triés selon leur nom et prénom (éventuellement par numéro lecteur) par ordre alphabétique. On peut avoir plusieurs lecteurs ayant le même nom et prénom, leur identifiant (numLecteur) sera différent.
  - Le numéro lecteur est un entier positif unique, fixe et définie automatiquement à la création d'un lecteur comme étant égale au numeroLecteur le plus grand avant l'insertion + 1.
  - Le tri se fait par fusion et la recherche est dichotomique.
  - Le champ « nbEmprunt » correspond au nombre d'occurrence du lecteur dans la liste d'emprunt et est automatiquement mis à jour.
  - Un lecteur ayant plus de 10 retards se voit attribuer une amende de 5 euros (amende non matérialisée, seulement énoncée) et son nombre de retard retourne à 0.
  - La suppression d'un lecteur est possible si et seulement si il n'a pas d'emprunt en cours.

- **Ouvrages :**
  - Les ouvrages sont enregistrés dans un fichier texte et chargés dans un tableau dynamique de pointeur vers des structures Ouvrage. La différence est que le nombre d'ouvrage est donné en début de fichier. Taille logique et physique sont alors les mêmes.
  - On trie les ouvrages par ordre alphabétique de leur cote qui est propre à chaque ouvrage (même des ouvrages de même titre).
  - Le tri se fait par échange et la recherche est dichotomique.
  - Son champ « emprunte » varie automatiquement en fonction de s'il se trouve ou non dans la liste d'emprunt.
  - La suppression d'un ouvrage est possible si et seulement si il n'est pas en cour d'emprunt.
  - La catégorie d'un ouvrage appartient à un ensemble fini, défini au préalable (BD, Fantaisie …).

- **Emprunt :**
  - Les emprunts sont lus et enregistrés dans un fichier binaire.
  - On ne connait pas le nombre d'emprunt et ceux-ci sont chargés dans une liste chainée composée d'un maillon (emprunt) et d'une adresse vers l'élément suivant. Pas de pointeur.
  - Le tri se fait par le numéro de lecteur de l'emprunteur, puis par cote.
  - Le tri et la recherche sont séquentiels.
  - La date de retour est par défaut égale au mois suivant l'emprunt. Modifiable.
  - Un emprunt peut être effectué si et seulement si le lecteur et l'ouvrage existe, si le lecteur a moins de 10 emprunts en cours et si l'ouvrage n'est pas déjà emprunté.
  - Un éventuel retard est pris en compte, une fois l'ouvrage rendu.

- **MENU :**
  - Un menu à choix multiple apparait dès le lancement du programme.
  - Un choix est demandé, vérifié et le « case » correspondant du switch est effectué.
  - A chaque action effectuée, une demande de continuer apparait. Si l'utilisateur répond « N », le programme se termine et les tableaux et la liste sont sauvegarder dans les fichiers d'origines.
  - Cas particulier : « case 13 ». Un nouveau menu avec des fonctionnalités de recherche supplémentaires s'ouvre.

V/Explication

Pour des raisons de longueur, seules les fonctions ambigües seront détaillées (les sources d'ambigüité sont soulignées).

1. void reduireLongueur(char *nom, char *nouvNom, int taille);

void ajouterBlanc(char *nom, int taille) ;

Problème à résoudre : Certains champs peuvent être très longs (ville, nom ouvrage…) ou trop courts et s'affichent mal dans les tableaux.

But : Ces deux fonctions permettent un alignement des éléments des tableaux et liste affichés.

Fonctionnement :

(Pour les occurrences trop longues) On créer, pour chacun des champs à afficher, une nouvelle chaine de caractère *nouvNom*, qui est une copie éventuellement plus courte de *nom*.

Cette chaine de caractères aura au maximum *taille* caractères. (PS : si *nom* a plus de *taille-2* caractères, deux petits points sont ajoutés à la fin de *nouvNom*).

(Pour les occurrences trop courtes) Ajouter autant de blanc à la fin de chaque champs qu'il faut pour que *nom* ait *taille* caractères.

2. Bool verificationNumTel(char *numTel);

Problème à résoudre : On souhaite éviter les fautes de frappe lors de la saisie d'un numéro de téléphone.

But : Tous les numéros de téléphone doivent avoir 10 chiffres et pas de lettre.

Fonctionnement : On parcourt *numTel* et on compte le nombre de chiffres. Si ce nombre est différent de 10 ou si un des caractères de *numTel* n'est pas un chiffre, on retourne **false**.

3. int comparerNomPrenom(char *nom1, char *prenom1, char *nom2, char *prenom2);

int comparerEmprunt(Emprunt e1, Emprunt e2);

int comparerDate(Date d1, Date d2);

Problème à résoudre : On souhaite comparer des structures dans des conditionnels.

But : Optimiser les recherches et les tris ou vérifier la validité d'un champ.

Fonctionnement : Similaire au « strcmp() », ces fonctions retournent un nombre négatif si (respectivement).

- *nom1* et *prenom1* sont situés avant *nom2* et *prenom2* par ordre alphabétique.
- *e1*.numLecteur < *e2.numLecteur* ou (*e1*.numLecteur = *e2.numLecteur* et *e1.cote* avant *e2.cote*).
- *d1* est antérieur à *d2*.

4. void sauvegarder(char *nomFichierLecteur, Lecteur **tabLect, int nbLect, char *nomFichierOuvrage, Ouvrage **tabOuv, int nbOuv, char *nomFichierEmprunt, ListeEmprunt l);

Fonctionnement : Se charge d'appeler les différentes fonctions de sauvegarde, afin de limiter la redondance au niveau du code.

5. int choixMenu(int nbPossibilite);

Problème à résoudre: éviter la redondance de code en écrivant plusieurs fois les mêmes fonctions.

But : Avoir une fonction qui peut correspondre à plusieurs menus ayant un nombre de choix différents.

Fonctionnement : Tant que le nombre saisie par l'utilisateur n'est pas compris entre 1 et *nbPossibilite* ou bien n'est pas égale à -1, l'utilisateur doit retaper.

6. void rechercher(Lecteur **tab, int nb, Ouvrage **tabOuv, int nbOuv, ListeEmprunt l);

Problème à résoudre : Plus de place à l'écran pour proposer de nouvelles fonctions sans devoir « scroller ».

But : Créer un sous-menu avec des fonctionnalités de recherche afin d'améliorer la visibilité à l'utilisateur et réduire le nombre de ligne de code dans la fonction global.

Fonctionnement : Si l'utilisateur choisi l'option 13, un sous-menu apparait. Le fonctionnement est le même que pour le Menu, mise à part qu'une seule fonction est exécutée avant de retourner au menu principal.

Les paramètres sont essentiels pour les différentes recherches proposées.