IFT1166 – TRAVAIL PRATIQUE #1 – 24 Septembre 1999

Gestion d'une base de données à partir d'une liste

Mohamed Lokbani

LA DATE LIMITE EST LE 17 OCTOBRE 1999 MINUIT, avec une pénalité de 2 points par jours de retard.

À faire seul ou en équipe de deux.

BUT

Ce TP a pour but de vous faire pratiquer les matières suivantes :

- Utilisation des outils de programmation Linux
- Un rappel des structures
- Programmation modulaire procédurale en C++
- Passage de paramètres par valeur ou par référence
- Fonctions surdéfinies
- L'allocation dynamique
- Les entrées sorties en C++

1. Description du problème

Nous aimerions gérer les dossiers de résultats des étudiants inscrits au cours IFT1166.

Le dossier d'un étudiant comporte son identité (nom et prénom), le programme auquel il est inscrit, son code permanent, les notes recueillies pour les travaux pratiques (tp1, tp2 et tp3), et pour les examens (intra et final) ; sa note finale (sur 100) et l'état de son passage au cours IFT1666 (succès ou échec).

Un dossier a l'allure suivante :

PAUL ALAIN LIBRE PAUA12106601 8 11.5 10.5 12 15 42 ECHEC

Ainsi PAUL (nom) ALAIN (prénom) est inscrit au programme LIBRE. Son code permanent est PAUA12106601. Ses notes sont : au tp1 8 (sur 10), au tp2 11.5 (sur 15) au tp3 12 (sur 15), la note d'intra 12 (sur 20) et le final 15 (sur 40). Sa note totale est de 42/100. Il a donc échoué à ce cours d'où ECHEC.

On suppose qu'un nom est composé d'un seul mot, de même pour le prénom.

Sur cette base de données, il vous est demandé d'implanter les fonctionnalités suivantes::

- ouvrir un nouveau dossier, dans le cas d'un étudiant nouvellement inscrit : pour cela, nous devons prévoir l'espace nécessaire pour stocker ce nouveau dossier. Permettre la saisie (via le clavier) des informations nécessaires pour compléter ce nouveau dossier. Effectuer la procédure des remises à jour des champs "note" et "etat", pour refléter le total réel et l'état de réussite ou d'échec à ce cours.

  Le calcul de la note totale d'un étudiant est effectué comme suit :

  Si (intra + final) ≥ 30 alors
  note = intra + final + tp1 + tp2 + tp3
  sinon
  note = intra + final + (tp1+tp2+tp3)/2

Un étudiant réussit le cours s'il arrive à obtenir plus de 50 dans la valeur totale de la note, et dans ce cas le champ "etat" sera « SUCCES » sinon il contiendra « ECHEC »
Le nouveau est inséré dans la liste selon l'ordre d'arriver.

- détruire un dossier, dans le cas d'un étudiant ayant retiré son inscription de ce cours : le nom de l'étudiant sera demandé (pour ne pas compliquer ce TP nous supposerons que les dossiers diffèrent par au moins le champ « nom »). Le dossier correspondant sera retiré de la base s'il existe. Et si c'est le cas, la liste est compactée pour récupérer son emplacement mémoire.

- consulter un dossier d'un étudiant donné : le nom de l'étudiant sera demandé. Le dossier correspondant sera affiché si ce nom existe dans la base.

- afficher la liste des étudiants qui ont réussi le cours : la liste des étudiants ayant réussi ce cours sera affichée par ordre croissant de nom.

- afficher la liste des étudiants qui ont échoué le cours : la liste des étudiants ayant échoué ce cours sera affichée par ordre croissant de nom.

- afficher des statistiques sur ce cours (nombre d'inscrits, taux de réussite et d'échec etc.) : Nous afficherons le nombre d'étudiants inscrits, le pourcentage de réussite et d'échec ainsi que la moyenne du groupe (sur 100).

  La moyenne du cours est la somme du champ "note" calculé pour chaque étudiant divisée par le nombre d'étudiants inscrits au cours.

- un affichage de la liste des inscrits triée par nom: par ordre croissant sur le champ "nom".

- un affichage de la liste des inscrits triée par note: par ordre décroissant sur le champ "note" (de la note la plus élevée vers la plus basse).

- un affichage de la liste originale à n'importe quel moment en respectant toujours l'ordre des inscrits (l'ordre établi lors de la création des dossiers dans la base de données).

Si la liste ne contient aucune élément, affichez "la liste est vide".

2. Travail demandé

Vous avez à écrire un programme en C++, permettant la gestion des dossiers du cours IFT1166.

Vous devez remettre deux fichiers :

- 'gliste.h' le fichier d'entête du programme.
- 'gliste.cpp' contenant les fonctions et le programme principal.

(s) dénote ce que le système va afficher comme réponse à votre requête.
(e) dénote les informations que vous entrez au clavier, pour être pris en compte par le programme.

Le programme démarre avec le menu suivant :

```
bdd cours ift1166

  1 - ajouter un etudiant a la liste
  2 - enlever un etudiant de la liste
  3 - afficher le dossier d'un etudiant donne
  4 - afficher la liste des recus, triee par noms
  5 - afficher la liste des recales, triee par noms
  6 - afficher des statistiques sur le cours
  7 - afficher la liste des inscrits triee par noms
  8 - afficher la liste des inscrits triee par notes
  9 - afficher la liste originale
```

```
 10 - quitter ce menu

Entrez votre choix:
```

Les fonctionnalités sont comme suit :

1 - ajouter un etudiant a la liste

```
(e) 1
(s)
(s) vous allez ouvrir un nouveau dossier et introduire les
informations necessaires :
(s)
(s) nom de l'etudiant & son prenom
(e) LEPAPE BERNARD
(s) programme d'etude
(e) LIBRE
(s) code permanent
(e) LEPB12052345
(s) note tp1 (sur 10), tp2 (sur 15) & tp3 (sur 15)
(e) 4 7 10
(s) note intra (sur 20) & note finale (sur 40)
(e) 10 20
(s) dossier LEPAPE BERNARD a ete cree
```

Remarquez que les entrées sont en majuscules. Le programme calcule lui-même la note finale et actualise le champ "etat" (ECHEC ou SUCCES) de l'étudiant.

Puis le système réaffiche le menu.

2 - enlever un etudiant de la liste

Deux cas peuvent se présenter:

```
(e) 2
(s)
(s) entrez le nom a retirer de la base
(e) PAUL
(s) dossier PAUL a ete retire
```

Puis le système réaffiche le menu.

Sinon

```
(e) 2
(s)
(s) entrez le nom a retirer de la base
(e) PAUL
(s) dossier de PAUL est inexistant
```

Puis le système réaffiche le menu.

3 - afficher le dossier d'un etudiant donné

Deux cas peuvent se présenter:

```
(e) 3
(s)
(s) entrez le nom a rechercher
(e) PAUL
(s) dossier de PAUL est inexistant
```

Puis le système réaffiche le menu

```
(e) 3
(s)
(s) entrez le nom a rechercher
(e) LEPAPE
(s)
(s) dossier de LEPAPE
(s)
(s) NOM PRENOM  PROGRAMME CODE TP1     TP2     TP3     INTRA
FINAL   TOTAL   ETAT
(s)
(s) LEPAPE  BERNARD LIBRE LEPB12052345    4       7       10
10      20      51 SUCCES
```

entre chaque champ, il a été inséré une tabulation

Puis le système réaffiche le menu.

4 - afficher la liste des recus, triee par noms

```
(e) 4
(s)
(s) Etudiant(s) ayant reussi au cours
(s)
(s) NOM PRENOM  PROGRAMME CODE TP1     TP2     TP3     INTRA
FINAL   TOTAL   ETAT
(s)
(s) LEPAPE  BERNARD LIBRE    LEPB12052345      4       7
10      10      20      51 SUCCES
(s) LORI    MARC    PHYSIQUE        LORM03059876    10      15
15      15      25 80     SUCCES
(s) MICHEL  NATHALIE        MATHEMATIQUE    MICN22610897  9
12.5    12.5    10 20      64      SUCCES
```

entre chaque champ, il a été inséré une tabulation

Puis le système réaffiche le menu.

5 - afficher la liste des recales, triee par noms

```
(e) 5
(s)
(s) Etudiant(s) ayant coule au cours
(s)
(s) NOM PRENOM  PROGRAMME MATRICULE TP1     TP2     TP3
INTRA FINAL   TOTAL   ETAT
(s)
(s) ABEL    SOPHIE  BIOLOGIE        ABEN07553675    3       4
5       6       7 19      ECHEC
```

entre chaque champ, il a été inséré une tabulation

Puis le système réaffiche le menu.

6 - afficher des statistiques sur le cours

```
(e) 6
(s)
(s) Les statistiques recueillies sur ce cours sont comme suit …
```

```
(s)
(s) Nombre d’inscrits : 50
(s) Taux de Reussite : 85%
(s) Taux d’Echec : 15%
(s) Moyenne du groupe : 75/100
```

Puis le système réaffiche le menu.

Les chiffres de ces statistiques sont cités à titre d'exemple.

7 - afficher la liste des inscrits triee par noms

```
(e) 7
(s)
(s) NOM PRENOM  PROGRAMME MATRICULE TP1     TP2     TP3
INTRA FINAL   TOTAL   ETAT
(s)
(s) ABEL    SOPHIE  BIOLOGIE        ABES25578432    3       4
5       6       7 19      ECHEC
(s) LEPAPE  BERNARD LIBRE   LEPB12052345    4       7       10
10      20      51 SUCCES
(s) LORI    MARC    PHYSIQUE        LORM03059876    10      15
15      15      25 80     SUCCES
(s) MICHEL  NATHALIE        MATHEMATIQUE    MICN22610897  9
12.5    12.5    10 20      64      SUCCES
```

entre chaque champ, il a été inséré une tabulation.

Puis le système réaffiche le menu.

8 - afficher la liste des inscrits triee par notes

```
(e) 8
(s)
(s) NOM PRENOM  PROGRAMME MATRICULE TP1     TP2     TP3
INTRA FINAL   TOTAL   ETAT
(s)
(s) LORI    MARC    PHYSIQUE        LORM03059876    10      15
15      15      25 80     SUCCES
(s) MICHEL  NATHALIE        MATHEMATIQUE    MICN22610897  9
12.5    12.5    10 20      64      SUCCES
(s) LEPAPE  BERNARD LIBRE   LEPB12052345    4       7       10
10      20      51 SUCCES
(s) ABEL    SOPHIE  BIOLOGIE        ABES25578432    3       4
5       6       7 19      ECHEC
```

entre chaque champ, il a été inséré une tabulation.

Puis le système réaffiche le menu.

9 - afficher la liste originale

```
(e) 9
(s)
(s) NOM PRENOM  PROGRAMME MATRICULE TP1     TP2     TP3
INTRA FINAL   TOTAL   ETAT
(s)
(s) MICHEL  NATHALIE        MATHEMATIQUE    MICN22610897  9
12.5    12.5    10 20      64      SUCCES
```

```
(s) LORI    MARC    PHYSIQUE        LORM03059876    10      15
15      15      25 80     SUCCES
(s) ABEL    SOPHIE  BIOLOGIE        ABES25578432    3       4
5       6       7 19      ECHEC
(s) LEPAPE  BERNARD LIBRE   LEPB12052345    4       7       10
10      20      51 SUCCES
```

entre chaque champ, il a été inséré une tabulation.

Puis le système réaffiche le menu.

10 - quitter ce menu

quitter complètement le programme.

3. Fichiers mis à votre disposition

En plus des deux fichiers que vous devez remettre, vous devez utiliser les fichiers suivants pour compléter la compilation de votre programme :

/u/dift1166/Sujet/tp1/libtp1.h

contient

la déclaration de la structure "eleve" et la définition de ses champs. Votre programme doit utiliser cette structure pour représenter un dossier d'étudiant.

Un prototype de fonction utilisée dans le fichier libtp1.o et permettant de lire des données et les recopier vers les champs d'une structure de données.

/u/dift1166/Sujet/tp1/libtp1.o

Un fichier objet, compilé sous Linux (**ne fonctionnera que sur Linux**).

Ce fichier contient une fonction permettant d'initialiser un tableau où chaque élément est du type "eleve" (structure). L'initialisation du tableau, vous permettra de réaliser la tache de saisie des données initiales de manière simple. D'un autre côté, elle va nous permettre de connecter à votre programme les données de test, lors de la correction de votre TP.

Cette fonction a le prototype suivant :

```
void load_data (eleve Etudiant[],int& numlistref);
```

où

Etudiant est un tableau où chaque élément est une structure du type "eleve"

"numlistref" est une référence à la taille du tableau après remplissage.

Notez bien que le tableau "Etudiant" a une taille maximale de 100 éléments (ne peut contenir plus de 100 éléments du type "eleve"). A vous de vérifier que la liste ne déborde pas de la taille préalablement fixée. **Il est à noter** que les champs "note" et "etat" doivent être mis à jour après le chargement des données i.e voir les explications "1. description du problème: ouvrir un nouveau dossier" au début de cet énoncé.

/u/dift1166/Sujet/tp1/resultat.txt

Un fichier texte montrant le fonctionnement du système, i.e les entrées/sorties, que devrait avoir votre programme.

Pour copier ces fichiers dans votre répertoire racine, faites comme suit:

cp /u/dift1166/Sujet/tp1/* ~

4. Compilation

-1- Avant de compiler votre programme inclure les lignes suivantes :

Dans votre fichier gliste.h

Dans la section des "include" :

#include “libtp1.h”

Dans votre fichier gliste.cpp

Un commentaire au début du fichier indiquant : vos noms, prénoms & login.

Dans la fonction "main", faites un appel de la fonction:

load_data (Etudiant,numlistref);

afin de remplir le tableau avec les données préparées pour vos tests.

-2-

Pour compiler, utilisez la commande:

g++ -Wall –o gliste gliste.cpp libtp1.o

les fichiers "libtp1.h" et "gliste.h" doivent se trouver dans le même répertoire que les fichiers "gliste.cpp" et "libtp1.o"

5. Remise

Vous devez faire deux types de remise :

- Papier (à glisser sous la porte de mon bureau, Pavillon André-Aisenstadt, local 2249)
  bien identifier sur le listing le cours et vos noms.

- Électronique : dans le répertoire où est votre programme, tapez:

remise ift1166 tp1 gliste.h gliste.cpp

LA DATE LIMITE EST LE 17 OCTOBRE 1999 MINUIT. On enlèvera 2 points par jours de retard.

6. Barème

Ce premier tp1 est noté sur 10 points. Nous vous conseillons de lire le contenu du fichier "resultat.txt"; sa lecture vous donnera une idée sur le fonctionnement réel du système.

Les points seront répartis comme suit :

Exactitude : 5 points

- un programme qui ne compile pas 0/5
- un programme avec des avertissements ("warnings") et qui donne les résultats escomptés 3.5
- un programme avec des avertissements "warnings" mais qui fait des choses non prévues dans la spécification 0/5

Modularité : 2 points

- structure de votre programme, décomposition entre .h et .cpp, décomposition en fonctions etc.

Efficience : 2 points

- La manière avec laquelle vous allez coder votre programme. Comment vous allez manipuler le tableau de données? L'espace mémoire utilisé etc.

Style : 1 point

- le code doit être lisible, bien indenté et commenté.

NOTE: L'utilisation de la fonction qsort de la librairie standard pour effectuer les tris, n'est pas permise.