Examen d'Algorithmique et Programmation Orientée Objet

2016-2017

1re année CSI Apprentissage Phelma - Semestre 1

Mars 2017

Première partie (30 mn) : Aucun document autorisé.

Deuxième partie (1h30) : Documents de COURS autorisés, documents de TP interdits.

Remarque importante

Les exercices de la partie 2 peuvent être traités indépendamment les uns des autres.
Le barème est indicatif.
Toutes les réponses doivent être données en utilisant le langage C++.

1 Question de cours et QCM — 30 minutes — SANS DOCUMENTS - 5 points

1. Deux erreurs sont faites dans la fonction C suivante.
   Expliquer ces erreurs et comment les corriger. Il n'est pas demandé de réécrire le code.

```
1  // Retourne un tableau contenant les 5 premiers nombres premiers
2  int * retourneTab_5_PremiersNombresPremiers(void) {
3    int tab[5];
4    //on range les nombres premiers dans tab
5    tab[1] = 1;
6    tab[2] = 2;
7    tab[3] = 3;
8    tab[4] = 5;
9    tab[5] = 7;
10
11   return tab;
12 }
```

2. Dans le programme qui suit, le prototype de la fonction `invert()` induit deux problèmes :
   — Le premier fait que le programme n'affiche pas ce qu'il faut, ligne 18.
   — Le second fait que l'appel de la fonction `invert` occupe un peu plus de mémoire, et est légèrement plus lent, que ce qu'on voudrait.

```
1  #include <iostream>
2  #include "Rational.h"
3  using namespace std;
4
5  // stocke dans r2 l'inverse de r1
```

```
6  bool invert( Rational r1, Rational r2 ) {
7    if( r1.getNum() == 0 ) {
8      cout << "erreur !" <<endl;
9      return false;
10   }
11   r2 = Rational ( r1.getDenom(), r2.getNum() );
12   return true;
13 }
14
15 int main() {
16   Rational a(10, 3), b(1,1) ;
17   invert(a, b);
18   cout << " b " << b; // doit afficher  le rationnel 3/10, mais il affiche 10/3
19   return 0;
20 }
```

Corrigez ces deux problèmes. Ne réécrivez pas tout le code, mais uniquement les lignes modifiées en indiquant le numéro de ligne.

3. Pourquoi est-il conseillé de déclarer `private` les attributs d'une classe ?
4. Que signifie le qualificateur `const` appliqué à une méthode, comme dans le cas suivant ?

```
class Truc {
  // ici les attributs...
  ...
  int uneMethode(int a, int b) const {
    ...
  }
  ...
};
```

5. On considère un programme C++ réparti dans 4 fichiers : trois fichiers sources `f1.cpp`, `f2.cpp`, `test.cpp` et trois fichiers header `f1.h` et `f2.h`. Le programme n'utilise aucune bibliothèque particulière.
   Donner la ligne de commande à exécuter dans le Terminal pour :
   — créer le fichier binaire `f1.o`
   — créer le programme exécutable.

2 Exercices — 1 heure et 30 minutes — documents de COURS autorisés - 15 points

2.1 Fonctions simples - 5 points

1. Écrire les fonctions suivantes :
   — Écrire la fonction `double & refMax(double val, double* tab, int nb)` qui retourne une référence sur le maximum des éléments du tableau tab de nb réels.
   — Écrire la fonction `int nbPositifs(double* tab, int nb)` qui retourne le nombre d'éléments positifs dans le tableau tab de nb réels. **Il est demandé d'utiliser un indice pointeur pour parcourir le tableau, et non pas un indice entier**.
   — Écrire la fonction `double * creerTableauValeurPositives(double t[ ], int n, int* pn2);` qui retourne un tableau de réels alloué dynamiquement, rempli avec les valeurs *positives* du tableau t de n éléments, et stocke dans $*pn2$ le nombre d'éléments du tableau retournée.
2. Écrivez un programme qui réalise les actions suivantes :
   — Lecture au clavier des éléments d'un tableau de 10 réels.
   — Remplacement du maximum du tableau par la valeur -5, sans utiliser de variable intermédiaire.
   — Création d'un nouveau tableau ne contenant que les valeurs positives du premier tableau
   — Libération du nouveau tableau

2.2 Compte bancaire - 10 points

Il s'agit d'écrire une classe C++ permettant de modéliser un compte bancaire, selon les spécifications suivantes :

1. Un compte bancaire est identifié par un *numéro de compte* (un entier). Ce numéro est déterminé automatiquement à la création du compte. Pour simplifier, les numéros de comptes vaudront un nombre entre 1 à n. Lorsque un nouveau compte est créé, Si (n) comptes qui ont déjà été créés, alors le numéro choisi pour le nouveau compte est $(n + 1)$.
2. Un compte est associé au nom de son *possesseur* (une chaine de caractères). Une fois le compte créé, le nom du titulaire du compte ne peut plus être modifié.
3. Le *solde* du compte (somme d'argent disponible sur un compte) est un nombre réel exprimée en Euros.
4. Le solde d'un compte peut éventuellement être négatif. Dans ce cas, on dit que le compte est à découvert. En aucun cas le solde d'un compte ne peut être inférieur à une valeur fixée pour ce compte, appelée le *découvert autorisé*. Le découvert autorisé peut varier d'un compte à un autre. Il est fixé à la création du compte et peut être modifié par la suite.
5. Le *taux d'intérêt* d'un compte, entier, est exprimé en pourcentage. Il est déterminé une fois pour toute à la création du compte.
6. *Appliquer le taux d'intérêt* consiste, si le solde est positif, à augmenter le solde du compte d'un montant égal à son solde initial multiplié par le taux d'intérêt en pourcent. Si le solde est négatif, appliquer le taux d'intérêt n'a pas d'effet.
7. Lors de la création d'un compte, seul l'identité du titulaire du compte est obligatoirement précisée. Le numéro du compte est choisi automatiquement. Le solde est toujours initialisé à 0. La valeur par défaut du découvert autorisé est de 1000 Euros et le taux d'intérêt par défaut est de 3%. Cependant, il est possible d'attribuer une autre valeur au découvert autorisé et au taux d'intérêt lors de la création du compte.
8. *Créditer* un compte consiste à ajouter un montant positif au solde du compte.
9. *Débiter* un compte consiste à retirer un montant positif au solde du compte. Le solde résultant ne doit en aucun cas être inférieur au découvert autorisé pour ce compte.

10. *Effectuer un virement* consiste à débiter un compte au profit d'un autre compte qui sera crédité du montant du débit.
11. Toutes les informations concernant un compte peuvent être consultées : numéro du compte, identité du titulaire, solde, montant du découvert autorisé, taux d'intérêt, situation du compte (est-il à découvert ?). Il est également possible de récupérer une représentation textuelle de l'état du compte.

Écrivez une classe `Compte` (fichier header `compte.h` et fichier source `compte.cpp`) correspondant aux spécifications précédentes.
Votre classe devra pouvoir fonctionner avec le programme principal donné sur la page suivante.

```
#include <iostream>
#include "compte.h"
using namespace std;

int main() {
  // le compte de "Gaston", decouvert autorise 0, taux d interet 4%
  Compte c1("Gaston", 0, 4) ;
  // le compte de "Cunegonde", decouvert autorise 100 Euros, taux d interet par defaut : 3%
  Compte c2("Cunegonde", 100) ;
  // le compte de "Gertrude", decouvert autorise par defaut (1000 Euros) et taux d interet par
    defaut (3%)
  Compte pc3 = new Compte("Gertrude") ;


  c1.crediter(100); // credite c1 de 100 Euros
  // c1.crediter( -100 );  // provoquerait message d erreur et sortie du programme : une somme
    creditee doit etre >=0

  c2 += 100 ; // credite c2 de 100 Euros
  c2 +=  -100 ; // credite c2 de 100 Euros // provoquerait message d erreur et sortie du
   programme : une somme creditee doit etre >=0

  pc3->crediter(100);

  // tous les comptes ont desormais un solde de 100 Euros.

  //Affiche numero, possesseur, solde, decouvert autorise et taux d interet des trois comptes.
  cout << "Les comptes : c1 : " << c1 << " . c2 : " << c2 << " . c3 : " << *pc3 << endl;

  c1.debiter(50); // reste 50 Euros.
  // c1.debiter(100); // provoquerait message d erreur et sortie du programme : solde
   insuffisant

  c1.setDecouvertAutorise(100); // nouveau decouvert autorise pour c1

  c1 -= 100 ; // debite c1 de 100 Euros. Solde negatif : -50 Euros.

  // c2 -=   (-100) ; // provoquerait message d erreur et sortie du programme : une somme
   debitee doit etre >=0

  c2.virerSur(&c1, 200); // solde de c1 devient +150 Euros. Solde de c2 devient -100 Euros.
  // c2.virerSur(&c1, 1000); // provoquerait message d erreur et sortie du programme : solde
   insuffisant sur c2

  c1.appliquerInteret(); // solde de c1 devient (150 + 150 * 4%) Euros

  cout << "infos de c1, une par une : " << endl;
  cout << "possesseur " << c1.getPossesseur() << endl;
  cout << "numero du compte " << c1.getNumero() << endl;
  cout << "solde " << c1.getSolde() << endl;

  if( c1.estADecouvert() ) {
    cout << "attention : c1 est a decouvert !" <<endl;
  }

  cout << "decouvert autorise " << c1.getDecouvertAutorise() << endl;
  cout << "taux d'interet  " << c1.getTauxInteret() << endl;

  delete pc3 ; // destruction du compte pointe par pc3
}
```