

Examen

Jeudi 16 Janvier 2013

Tous documents autorises (sauf copie du voisin, appareillage electronique, etc). Les telephones portables, comme tout autre moyen de communication vers l'exterieur, doivent être eteints. Le temps a disposition est de trois heures. Motivez bien vos reponses.

1 Exercice

1. Indiquer (en justifiant) ce que ce programme affiche lors de son execution.
2. A quoi sert le mot-cle `virtual` devant la definition des destructeurs? Donner un exemple permettant d'illustrer sa necessite.

```
#include <iostream>
using namespace std;
class Premiere {
public:
    Premiere()          { cout << "ctor premiere" << endl; }
    virtual ~Premiere() { cout << "dtor premiere" << endl; }
    void salut()         { cout << "salut" << endl; }
    virtual void bonjour() { cout << "bonjour" << endl; }
};
class Seconde : public Premiere {
public:
    Seconde()          { cout << "ctor seconde" << endl; }
    virtual ~Seconde() { cout << "dtor seconde" << endl; }
    void salut()         { cout << "salutations" << endl; }
    virtual void bonjour() { cout << "bien le bonjour" << endl; }
};
int main() {
    Premiere p;
    Seconde *s = new Seconde;
    p.salut(); p.bonjour();
    (*s).salut(); (*s).bonjour();
    Seconde *ps = s;
    ps->salut(); ps->bonjour();
    Premiere *pp = ps;
    pp->salut(); pp->bonjour();
    delete s;
    return 0;
}
```

2 Exercice

Soit l'extrait de programme suivant :

```
void quickSort(int arr[], int left, int right) {
    int i = left, j = right; int tmp; int pivot = arr[(left + right) / 2];

    /* partition */
    while (i <= j) {
        while (arr[i] < pivot) i++;
        while (arr[j] > pivot) j--;
        if (i <= j) {
            tmp = arr[i]; arr[i] = arr[j]; arr[j] = tmp;
            i++;
            j--;
        }
    }
    /* recursion */
    if (left < j) quickSort(arr, left, j);
    if (i < right) quickSort(arr, i, right);
}
```

On souhaite (pour certaines donnees experimentales) compter le nombre de certaines classes d'operations effectuees durant l'execution de la fonction `quickSort` (mesure experimentale de la complexite algorithmique).

1. Transformer/adapter le code donne de sorte que l'on puisse compter diverses operations (par exemple les tests et/ou les affectations). Attention : car la seule modification acceptee dans le code precedent est le remplacement de certaines types ; aucune instruction supplementaire ne doit être retiree ou ajoutee. Bien entendu, il est peut-être necessaire de construire par ailleurs des types particuliers (si c'est le cas, il faudra en fournir le code).

3 Exercice

1. Ecrire un type `List<T>` permettant de represente une structure d'elements (du même type) et chaînes (a la construction d'un element `List<T>`, la liste sera vide, et l'on disposera de 4 methodes : `add(T&)`, `T& remove(i)`, `bool isIn(T&)` et `T get(i)` permettant respectivement d'ajouter un element en fin de liste, de supprimer un element de la liste etant donne sa position dans la liste, de tester si un element est dans la liste et de recuperer l'element d'indice donne.
2. Puisqu'il s'agit d'un modele de classe, preciser les contraintes imposees sur le type pour que l'instanciation du modele fonctionne.
3. Ecrire un iterateur de cette structure, un tel iterateur devra pouvoir être obtenu par une instruction du genre `List::iterator i = l.getIterator();` ou `l` est une liste instanciee. Les methodes disponibles sur un iterateur sont `getCurrent` et `next` qui, respectivement, renvoient l'element courant de l'iteration en cours et deplace le « curseur » vers l'element suivant (s'il y en a un) et renvoi un boolean pour indiquer qu'il y a bien un element courant.

4 Problème

La phylogenie nous indique que les Bovins (qui ont un cuir d'une certaine epaisseur) et les Ovins (qui sont couverts de laine d'une certaine longueur) sont des Ruminants (qui ruminent) eux-même des Mammiferes (qui allaitent leurs petits). Parmi les Bovins on trouve les Bœufs dont la representante la plus connue est Marguerite, jolie vache laitiere de 564kg et elevee par Marcel et Germaine Delapeau, fermiers a Brive-la-gaillarde. Pour information, les Vaches sont des Bœufs femelles.

1. Comment traduit-on en UML une telle specification ?
2. Placer dans les classes identifiées les caracteristiques/operations sous-entendues par la specification.
3. Traduire chaque classe UML en une classe C++ correspondante avec attributs et methodes. Prendre soin de specifier tous les modificateurs adequats, necessaires ou utiles (const, etc).
4. Ecrire un programme C++ permettant de creer une version numerique de la charmante Marguerite et de ses proprietaires et d'ajouter leurs caracteristiques.
5. Ecrire une usine (factory) permettant de creer un Bœuf de sexe, nom et race quelconques (sous la forme d'attributs). Modifier la classe Vache en consequence. Modifier la construction de Marguerite afin d'obtenir une Vache de race Limousine.
6. Si l'on souhaite obliger l'utilisateur a passer par l'usine pour obtenir une Vache, que doit-on modifier ?
7. Si l'on decide de distinguer les races de Bœufs a l'aide de classes differentes et non plus d'attributs, qu'est-ce que cela change ? Ecrire les classes Limousine et Normande.
8. Sachant qu'une Limousine fait « Meuheuh » et une Normande fait « Meeeuuhhh » juste apres avoir mange, implanter ce mecanisme dans les classes adequates. Attention, les Ovins eux, ne font aucun bruit particulier apres avoir mange...