Université du Luxembourg 2004–2005

D.U.T. Informatique 2ème année mercredi 3 novembre

Programmation avancée en langage C++
Examen oral

Sebastien.Varrette@imag.fr (Bureau BR.4.07)

Durée: 30 minutes

Tout document interdit. Les exercices suivants (hormis l'exercice 1) sont à préparer sur machine avant d'être présentés et justifiés aux correcteurs.

1 Contrôle de connaissances

On considère les programmes C++ suivants :

```
#include <iostream>
using namespace std;

void reset(int i) { i = 0; }

int main() {
    int n = 14;
    reset(n);
    cout << n << endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;

int main() {
    int * p;
    *p = 2;
    cout << *p << endl;
    return 0;
}
```

1. Quel est le résultat de l'exécution du programme de gauche ?
2. Même question pour le programme de droite.
3. En C++, qu'est ce qu'un constructeur ? Comment est il identifié ?
4. On considère le programme C++ suivant composé des fichiers suivants :

Fichier `A.h`

```
class A {
  private:
    int _x;
  public:
    A(int x = 0);
};
```

Fichier `A.cpp`

```
#include "A.h"

A::A(int x) : _x(x) {}
```

Fichier `B.h`

```
class B : public A {
  private:
    int _y;
  public:
    B(int x=0, int y=0);
};
```

Fichier `B.cpp`

```
#include "A.h"
#include "B.h"

B::B(int x, int y) {
    _x = x;
    _y = y;
}
```

Fichier `main.cpp`

```
#include "A.h"
#include "B.h"

int main() {
    return 0;
}
```

(a) Pourquoi la compilation de ce programme provoque t il une erreur ?

(b) Proposez une solution pour résoudre ce problème.

2 STL et tableaux dynamiques

On souhaite construire une classe en C++ pour manipuler un ensemble d'entiers. On part de la déclaration suivante :

```
#include <iostream>
#include <list>

using namespace std;

class IntegerList {
  private:
    /* A COMPLETER */
  public:
    IntegerList();     // Constructeur
    ~IntegerList();    // Destructeur
    size_t cardinal(); // nombre d'éléments de la liste
    void operator< (int e);   // pour ajouter un entier ds la liste;
                                        // Ex: L < 3 ajoute 3 dans la liste L
          // ATTENTION : insertion uniquement si e n'est pas déjà ds la liste:
    bool operator== (IntegerList & L); // Teste l'égalité de 2 ensembles
    void print();
};
```

ATTENTION : on cherche à représenter des ensembles d'entiers aussi chaque élément ne peut être présent plus d'une fois.
L'objectif de cet exercice est de fournir le corps de cette classe dans deux contextes différents :

1. En utilisant la STL et plus particulièrement la classe `list` (voir §2.1).
2. En utilisant des pointeurs (§2.2).

2.1 Utilisation de la STL

1. Quelle donnée membre faut-il prévoir ?
2. Donnez le code du constructeur.
3. Donnez le code du destructeur.
4. Donner le code de la fonction `cardinal`
5. Donner le code de la fonction `print`
6. Donner le code de l'opérateur `<`. On prendra garde à ne pas insérer de doublons.
7. Donner le code de l'opérateur `==`.

2.2 Utilisation de pointeurs

On souhaite cette fois ci utiliser une approche plus proche du langage "C" en utilisant des pointeurs pour représenter un ensemble d'entiers.

1. Quelles données membres faut-il prévoir ?
2. Donnez le code du constructeur.
3. Donnez le code du destructeur.
4. Donner le code de la fonction `cardinal`
5. Donner le code de la fonction `print`
6. Donner le code de l'opérateur `<`. On prendra garde à ne pas insérer de doublons.
7. Donner le code de l'opérateur `==`.