Trimestre Hiver, 2006

Mohamed Lokbani

IFT1169 – Examen Intra –

Inscrivez tout de suite votre nom et code permanent.

Nom: _____| Prénom(s): _____|

Signature: _____| Code perm: _____|

Date : mardi 21 février 2006

Durée : 2 heures (de 18h30 à 20h30)

Local : Z-110 ; Pavillon Claire-McNicoll (ancienne aile Z).

Directives:

- Toute documentation permise.
- Calculatrice **non** permise.

- Répondre directement sur le questionnaire.
- Les réponses **doivent être brèves, précises et clairement présentées.**

1._____/10 (1.1, 1.2)

2._____/15 (2.1, 2.2)

3._____/25 (3.1, 3.2)

4._____/25 (4.1, 4.2, 4.3, 4.4)

5_____ /25 (5.1, 5.2, 5.3)

Total:_____/100

Directives officielles

* Interdiction de toute communication verbale pendant l'examen.

* Interdiction de quitter la salle pendant la première heure.

* L'étudiant qui doit s'absenter après la première heure remet sa carte d'étudiant au surveillant, l'absence ne devant pas dépasser 5 minutes.

* Un seul étudiant à la fois peut quitter la salle.

* Toute infraction relative à une fraude, un plagiat ou un copiage est signalée par le surveillant au directeur de département ou au professeur qui suspend l'évaluation.

F.A.S

Exercice 1 (10 points)

1.1 Tout en justifiant votre réponse, quels sont les constructeurs qui doivent exister dans la classe B pour que le fragment de code suivant puisse compiler correctement ?

```
class A: public B {
public:
   A(int j) {}
   A(int j, int k):B(j,k) {}
};
```

1.2 Tout en justifiant votre réponse, dites combien de fois est appelé le constructeur de la classe « A » dont la signature est « `A()` » dans le fragment de code suivant :

```
A u,v,*x;
x=new A();
A y=u;
A *z=new A();
```

Exercice 2 (15 points)

2.1 Dans cet exercice, on suppose que la variable « a » est du type « int », et qu'elle peut prendre n'importe quelle valeur positive ou nulle de ce type. Changer l'instruction suivante pour une instruction plus simple qui a exactement le même comportement.

```
if ( ((a%2)==0) || ((a%2)==1) ) cout << "IFT\n";
else cout << "1169\n";
```

Réponse (avec une courte explication) :

2.2 Réécrire le fragment de code suivant par un fragment de code plus simple qui a exactement le même comportement. Vous ne devez utiliser pour cela que les méthodes de la classe « string ».

```
void f( string &s1, string &s2, string &s3){
    const char *cs = (s1+s2).c_str();
    cout << cs;

    if ( strlen(cs = (s2+s3).c_str()) < 8 && cs[0] == 'a')
        cout << cs;
}
```

Réponse (avec une courte explication) :

Exercice 3 (25 points)

3.1 En supposant que la variable « x » est du type « int », que vous devez écrire un bloc « catch » pour chaque type d'exception levée et que vous pouvez supposer que toutes ces exceptions ont déjà été définies ; écrire les différents blocs « try-catch-finally » pour ce qui suit (toutes les instructions doivent figurer dans un seul fragment de code):

- Si « x » vaut 42, lever l'exception « A ».
- Si « x » vaut 44, lever l'exception « B » (dérive de l'exception « A »).
- Si « x » vaut 46, lever l'exception « C » (dérive de l'exception « B »).
- Si « x » vaut 48, lever l'exception « D » (dérive de l'exception « A »).
- Si « x » > 50, lever l'exception « E » (dérive de l'exception « D »).
- Dans le bloc « catch » qui gère l'exception « A », lever l'exception « E ».
- Dans le bloc « catch » qui gère l'exception « D », faite un « return ».
- Le bloc « finally », affiche en sortie la valeur de « x » et lève l'exception « F » (dérive de « A »).

3.2 Dérouler votre fragment de code pour une valeur de « x » égale à « 48 ».

Exercice 4 (25 points)
Soit le programme suivant :

```
1   #include <iostream>
2   #include <string>
3   using namespace std;
4
5   class DicoBasic {
6   public:
7       DicoBasic(const int N) {
8           tableau = new string[N];
9       }
10      ~DicoBasic() {
11      }
12      const string operator[](const int index) {
13          return tableau[index];
14      }
15  private:
16      string* tableau;
17      const int N = 25;
18      DicoBasic(const DicoBasic&); // Voir "Question 4.1"
19  };
20  int main() {
21      DicoBasic mt(10);
22      mt[2] = "ift1169";
23      cout << "mt[2]: " << mt[2] << endl;
24      return 0;
25  }
```

4.1 Quelles sont les conséquences de la déclaration située à la ligne « 18 » ?

4.2 Le programme contient **deux erreurs** qui empêcheront sa compilation, trouvez-les et corrigez-les (dans le code).

Erreur 1, Ligne _____, Explication & Correction :

Erreur 2, Ligne _____, Explication & Correction :

4.3 Après avoir corrigé les deux erreurs, le code suivant pose toujours un problème :

```
int main() {
    const int N = 1024*1024*10;
    for(int k = 0; k < 1000; k++) {
        DicoBasic m(N);
    }
    return 0;
}
```

Expliquer le problème et comment le corriger.

4.4 Après avoir répondu aux questions 4.2 et 4.3, tout en justifiant votre réponse, est-ce que vous pensez que cette classe est maintenant sécuritaire ? On dit qu'une classe est sécuritaire si elle ne peut pas causer le plantage du programme. Si la réponse est positive, confirmer qu'elle est sécuritaire en expliquant cela par une phrase ou deux. Si la réponse est négative, proposez au moins une modification permettant de rendre la classe « DicoBasic » plus sécuritaire. Pour cette question, on peut supposer qu'il n'y aura pas appel à l'opérateur d'affectation ni au constructeur de recopie.

Exercice 5 (25 points)

Les produits céréaliers suivants sont tous à base de blé:

- Le pain est à base de farine ou de son.
- Les gâteaux peuvent être à base de farine ou de semoule.
- Les pâtes ou le couscous sont à base de semoule.

Ainsi donc le blé va subir d'abord diverses transformations Au commencement il sera subdivisé en deux catégories lors de la récolte : blé tendre et blé dur. Le blé tendre sera transformé en farine ou en son ; alors que le blé dur sera transformé en semoule.

5.1 Si la classe de base est une classe abstraite pure dont le nom est « Céréale », et si le « pain », les « gâteaux », les « pâtes » et le « couscous » font parti eux aussi (comme classe de base ou classe dérivée), parmi d'autres classes, du schéma d'héritage complet ; **dessiner ce schéma d'héritage, tout en expliquant votre démarche**. Sur votre schéma d'héritage, mettre en évidence toutes les classes de base, classes dérivées, les dérivations simples et les dérivations multiples (s'il y'en a).

5.2 Si vous avez utilisé dans votre schéma d'héritage --l'héritage multiple--, citer les problèmes qu'il peut causer. Citer aussi comment les corriger.

5.3 Si vous avez utilisé dans votre schéma d'héritage --l'héritage multiple--, est-ce qu'il est possible de le transformer en --héritage simple--? Si oui, comment?