Trimestre Hiver, 2007

Mohamed Lokbani

IFT1169 – Examen Final –

Inscrivez tout de suite votre nom et le code permanent.

Nom: _____| Prénom(s): _____|

Signature: _____| Code perm: _____|

Date : jeudi 19 avril 2007

Durée : 3 heures (de 18h30 à 21h30)

Local : Z-317 ; Pavillon Claire McNicoll.

Directives:

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

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

1._____/15 (1.1, 1.2, 1.3)

2._____/15 (2.1, 2.2, 2.3)

3._____/20 (3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8)

4._____/15 (4.1)

5._____/15 (5.1)

6._____/20 (6.1)

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 remettra 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 (15 points)

1.1 Expliquez brièvement le sens de la phrase suivante : « l'amitié n'est ni héritée ni transitive ni réciproque ».

1.2 Tout en expliquant brièvement votre réponse, doit-on utiliser plutôt des fonctions membres ou des fonctions amies ?

1.3 Soit les opérateurs surchargés de la classe suivante :

```
class A{
      friend A operator+ (const A& a1, const A& a2);
public:
      A operator* (const A& a1);
};
```

Donnez l'implémentation correcte, définie à l'extérieur de la classe, des opérateurs mentionnés dans « A »?

		correcte	incorrecte
a)	`A operator+ (const A& a1, const A& a2)` `{ A tmp; /*... code de l'opérateur...*/ return tmp; }`		
Pourquoi ?			
b)	`A* A::operator* (const A& a)` `{ A tmp; /*...code de l'opérateur...*/ return &tmp; }`		
Pourquoi ?			
c)	`A A::operator* (const A& a1, const A& a2)` `{ A tmp; /*...code de l'opérateur...*/ return tmp; }`		
Pourquoi ?			
d)	`A operator+ (const A& a1)` `{ A tmp; /*...code de l'opérateur...*/ return tmp; }`		
Pourquoi ?			
e)	`A A::operator* (const A& a1)` `{ A tmp; /*...code de l'opérateur...*/ return tmp; }`		
Pourquoi ?			

Exercice 2 (15 points)

2.1 Deux instances de la même classe générique sont considérées comme des sous-classes découlant l'une de l'autre. Est-ce vrai ?

2.2 Transformez la fonction suivante en une fonction générique permettant de supporter n'importe quel type. Assurez-vous que la fonction générique puisse fonctionner aussi bien sur des instances de classes qui implémentent l'opérateur « + » de manière que le polymorphisme dynamique d'inclusion (associé aux fonctions vs. ligature dynamique pour les classes) puisse avoir lieu quand cela est disponible.

```
int somme(int a, int b, int c){
     return a+b+c;
}
```

2.3 En tenant compte de la fonction générique « somme » définie en « 2.2 », définissez le strict minimum de la classe générique « UnObj » paramétrée par un seul type « T » pour que le code suivant puisse fonctionner correctement :

```
UnObj<int> s1, s2;
UnObj<int> s3 = somme(s1,s2,s2);
```

Par le terme « définir », on entend par cela l'écriture du code complet des méthodes nécessaires de la classe générique « UnObj ».

Exercice 3 (20 points) Complétez le programme suivant :

```
#include<iostream>
#include<MYSTERE1>
```

3.1 Tout en justifiant votre réponse, donnez le nom du fichier d'inclusion qui doit prendre la place de « **MYSTERE1** » ?

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

void Truc(vector<char>& v){
    for(unsigned int i=0;i<MYSTERE2;i++)
```

3.2 Tout en justifiant votre réponse, donnez l'instruction qui doit prendre la place de « **MYSTERE2** » ?

```
        cout<<v.at(i);
    cout << endl;
}

int main () {

    MYSTERE3 UnVec;
```

3.3 Tout en justifiant votre réponse, donnez le type de la variable « UnVec » ?

```
    char c;
    int i;
    for(i=3;i<8;i++){
        c = static_cast<char>('a' + i);
        UnVec.push_back(c);
    }
    Truc(UnVec);
```

3.4 Tout en justifiant votre réponse, donnez l'affichage en sortie ?

```
        UnVec.insert(UnVec.begin()+2,'z');
        Truc(UnVec);
```

3.5 Tout en justifiant votre réponse, donnez l'affichage en sortie ?

```
        UnVec.erase(UnVec.end()-2);
        Truc(UnVec);
```

3.6 Tout en justifiant votre réponse, donnez l'affichage en sortie ?

```
        // Rappel: « accumulate » est une version généralisée de
        // la sommation.
        // « accumulate(arg1,arg2,arg3) » initialise le résultat
        // avec la valeur du arg3 et en ajoute la valeur de chacun
        // des éléments de l'ensemble défini dans l'intervalle
        // [arg1,arg2).
        char ch = accumulate(UnVec.begin(), UnVec.begin(), 'a');
        cout << ch << endl;
```

3.7 Tout en justifiant votre réponse, donnez l'affichage en sortie de la variable « ch » ?

```
        // Rappel: count_if(arg1,arg2,arg3) compte le
        // nombre d'éléments X qui vont satisfaire la condition décrite
        // par (arg3) dans l'intervalle [arg1, arg2)

        int j  =
                  count_if(UnVec.begin(), UnVec.end(),
                              bind2nd(greater_equal<char>(), 'e'));

        cout << j << endl;
```

3.8 Tout en justifiant votre réponse, donnez l'affichage en sortie de la variable « j » ?

```
        return 0;

    }
```

Exercice 4 (15 points) Expliquez brièvement les instructions de la méthode « main ». Par ailleurs donnez un exemple du contenu du fichier « essai.txt » et la sortie correspondante.

```
01  #include <iostream>
02  #include <list>
03  #include <map>
04  #include <fstream>
05
06  using namespace std;
07  int main() {
08
09      ifstream source("essai.txt");
10      string s = "";
11      typedef map<int, list<string> > untype;
12      untype m;
13      while (source >> s)
14          m[s.size()].push_back(s);
15      for (untype::iterator p=m.begin(); p!=m.end(); ++p) {
16          cout << p->first;
17          for (list<string>::iterator q=p->second.begin();
18                                              q!=p->second.end(); ++q)
19              cout << " " << *q;
20          cout << endl;
21      }
22      cout << endl << "Fin du test !" << endl;
23      return 0;
24  }
```

Exercice 5 (15 points) En expliquant brièvement, dites ce que va afficher en sortie le programme suivant qui compile et qui s'exécute correctement. Le fichier « test.txt » contient le texte suivant :

```
Passez de bonnes vacances.
```

```
#include <fstream>

using namespace std;

int main() {
    char ch1='2', ch2='1';
    char str1[10] = {'1','2','3','4','5','6','7','8','9','\0'};
    char str2[10] = {'1','2','3','4','5','6','7','8','9','\0'};
    char str3[10] = {'1','2','3','4','5','6','7','8','9','\0'};
    ifstream infile("test.txt");
    infile >> str1;
    cout << "str1: " << str1 << endl;
```

sortie :

```
    infile.get(ch1);
    cout << "ch1: " << ch1 << endl;
```

sortie :

```
    infile.getline(str2, 5, '\n');
    cout << "str2: " << str2 << endl;
```

sortie :

```
    infile.getline(str3, 5, ' ');
    cout << "str3: " << str3 << endl;
```

sortie :

```
    infile.get(ch2);
    cout << "ch2: " << ch2 << endl;
```

sortie :

```
    return 0;
}
```

Exercice 6 (20 points) Ecrivez un programme qui ouvre un fichier en entrée, passé en argument, et qui calcule la somme des digits extraits du fichier affichant ainsi le résultat en sortie.

Par exemple, si le fichier d'entrée est « test.txt » et contient le texte suivant :

```
34a5
8def
```

L'affichage en sortie obtenu sera comme suit :

```
La somme totale de 3+4+5+8 est 20
```

Assurez-vous d'inclure des messages instructifs en cas d'erreurs (argument manquant, fichier introuvable, erreur de lecture etc.).

Bonnes vacances !