Trimestre Automne, 2016

Mohamed Lokbani

IFT1169 – Examen Final –

Inscrivez tout de suite : votre nom et le code permanent.

Nom : _____| Prénom(s) : _____|

Signature : _____| Login : _____|

Date : Vendredi 9 décembre 2016

Durée : 3 heures (de 16h30 à 19h30)

Local : Z-310, Pavillon Claire McNicoll

Directives :

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

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

1._____/20

2._____/20

3._____/15

4._____/15

5._____/15

6._____/15

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 (20 points) ce programme génère une cascade d'erreurs à la compilation. On vous demande de le corriger tout en respectant le principe d'encapsulation des données et l'affichage désiré en sortie. Vous pouvez ajouter de nouvelles lignes et corriger ou réécrire celles qui posent problème.

```
 1  #include <stdlib.h>
 2  #include <iostream>
 3
 4  template <class T> class Pair {
 5  public:
 6          Pair(T a, T b): first_(a), second_(b) { }
 7          void Print() {
 8                  cout << "(" << first_ << "," << second_ << ")" << endl;
 9          }
10          void Set(T a, T b) { first_ = a; second_ = b; }
11  private:
12          T first_, second_;
13  }
14
15  int main(int argc, char **argv) {
16          Pair<string> stringpair("Hello", "World");
17          Pair<string> pairarr = new Pair[2];
18
19          pairarr[0].first_ = "jour";
20          pairarr[0].second_ = "nuit";
21          pairarr[1].Set("plein", "vide");
22          stringpair.Print();
23          pairarr[0]->Print();
24          pairarr[1]->Print();
25          delete pairarr;
26
27          return 0;
28  }
```

Affichage désiré en sortie

(Hello,World)
(jour,nuit)
(vide,plein)

Expliquez brièvement votre démarche

Exercice 2 (20 points) ce programme compile et s'exécute correctement.

```
 1  #include <iostream>
 2  using namespace std;
 3  class Base {
 4  public:
 5          Base() { cout << "Base::cons" << endl; }
 6          void p() { cout << "Base::p" << endl; }
 7          virtual void q() { cout << "Base::q" << endl; }
 8          void operator=(Base &rx) { cout << "Base::=" << endl; }
 9  };
10
11  class Derivee : public Base {
12  public:
13          Derivee() { cout << "Derivee::cons" << endl; }
14          void p() { cout << "Derivee::p" << endl; }
15          void q() { cout << "Derivee::q" << endl; }
16          void operator=(Derivee &rx) { cout << "Derivee::=" << endl; }
17  };
18
19  int main(int argc, char **argv) {
20          Base base1, base2, *baseptr;
21          Derivee derivee1;
22          base2 = base1;
23          base2.p();
24          base2.q();
25          baseptr = (Base *) &derivee1;
26          baseptr->p();
27          baseptr->q();
28          derivee1.p();
29          derivee1.q();
30          base1 = derivee1;
31          return 0;
32  }
```

Que va afficher le précédent programme? Expliquez brièvement votre démarche.

Exercice 3 (15 points) ce programme compile et s'exécute correctement.

```
 1  #include <iostream>
 2
 3  using namespace std;
 4
 5  template<typename T> void Affiche(T x) { std::cout << x; }
 6
 7  class X {
 8  public:
 9         X() { Affiche('0');}
10         ~X() { Affiche(1);}
11  };
12
13  int main() {
14         try {
15                Affiche(5);
16                throw X();
17                Affiche(6);
18         } catch (X e) {
19                Affiche(7);
20         }
21         Affiche(8);
22
23         return 0;
24  }
```

Que va afficher le précédent programme? Expliquez brièvement votre démarche.

Exercice 4 (15 points) ce programme compile et s'exécute correctement.

```
 1  #include <iostream>
 2
 3  template<typename T> void Affiche(T x) { std::cout << x; }
 4  int main() {
 5          Affiche(0);
 6          auto f = []{Affiche(4);};
 7          auto g = [f](){Affiche(2);f();Affiche(3);};
 8          Affiche(9);
 9          g();
10
11          return 0;
12  }
```

Que va afficher le précédent programme? Expliquez brièvement votre démarche.

Exercice 5 (15 points) ce programme compile et s'exécute correctement.

```
 1  #include <iostream>
 2  #include <memory>
 3
 4  using namespace std;
 5  struct Noeud {
 6          int* val_;     // ptr vers une donnée du noeud
 7          Noeud *suiv_;  // le noeud suivant dans la liste sinon nullptr
 8  };
 9
10  int main() {
11          Noeud *liste = new Noeud();
12          liste->val_ = new int(17);
13          Noeud *p = new Noeud();
14          p->val_ = new int(42);
15          p->suiv_ = nullptr;
16          liste->suiv_ = p;
17
18          // affichage de la liste
19          for (auto n = liste; n != nullptr; n = n->suiv_)
20                  cout << *(n->val_) << " ";
21          cout << endl;
22          return 0;
23  }
```

En utilisant l'utilitaire « valgrind » à l'aide de la commande « valgrind ./exo5 », il nous signale une fuite de mémoire :

```
==11565== HEAP SUMMARY:
==11565==       in use at exit: 40 bytes in 4 blocks
==11565==   total heap usage:  6 allocs, 2 frees, 73,768 bytes allocated
==11565==
==11565== LEAK SUMMARY:
==11565==    definitely lost: 16 bytes in 1 blocks
==11565==    indirectly lost: 24 bytes in 3 blocks
==11565==      possibly lost:   0 bytes in 0 blocks
==11565==    still reachable:   0 bytes in 0 blocks
==11565==         suppressed:   0 bytes in 0 blocks
```

5.1/ expliquez qui est responsable de cette fuite de mémoire?

5.2/ en n'utilisant que les pointeurs intelligents « unique_ptr » et « shared_ptr », corrigez le précédent programme pour permettre d'éliminer la fuite de mémoire. Le but est d'utiliser les pointeurs intelligents afin de remplacer les pointeurs classiques (« * »). Vous allez en conséquence devoir ajuster votre structure et votre programme. Le programme doit réaliser la même tâche, créer une liste de nœuds, les initialiser puis les afficher en sortie.

Exercice 5 (15 points)

On vous demande d'écrire la fonction « C++ » « intersect », dont la signature est :

```
void intersect(const list<string> &lst1, const list<string> &lst2, list<string> &rep)
```

Cette fonction cherche les éléments communs aux deux listes « lst1 » et « lst2 » et stocke le résultat dans le 3e argument « rep ».

Voici un exemple d'une fonction main qui utilise une telle fonction :

```
 1 int main(){
 2     list<string> l1 = {"cerise","pomme", "banane", "citron","mangue", "orange", "melon"};
 3     list<string> l2 = {"pamplemousse","citron","raisin","orange"};
 4     list<string> resultat = {"asperges", "broccoli"};
 5
 6     intersect(l1,l2,resultat);
 7
 8     for (auto i: resultat)
 9                cout << i << endl;
10
11     return 0;
12 }
```

L'affichage obtenu en sortie est :

```
citron
orange
```

Notez qu'un élément ne peut se trouver qu'une fois dans une liste (un seul exemplaire, pas de duplication).

Lors de la correction, je vais tenir compte de la clarté de votre code. Expliquez brièvement votre démarche.

Joyeuses fêtes et bonne année!