Trimestre Automne, 2022

Mohamed Lokbani

IFT1169 – Examen Final –

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

Nom : _____| Prénom(s) : _____|

Signature : _____| Login : _____|

Date : vendredi 16 décembre 2022

Durée : 3 heures (de 16h30 à 19h30)

Local : Z-310, Pavillon Claire McNicoll

Directives :

- Toute documentation est permise.
- Appareils électroniques **non** permis.

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

1._____/20 (1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9. 1.10)

2._____/15 (2.1, 2.2, 2.3, 2.4, 2.5)

3._____/15 (3.1)

4._____/15 (4.1, 4.2, 4.3)

5._____/15 (5.1, 5.2, 5.3, 5.4)

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 (20 points) répondez par « vrai » ou « faux » en y incluant une très courte explication.

1.1 [VRAI | FAUX] Les « wxWidgets » est un des thèmes enseignés dans le cours IFT1169.

1.2 [VRAI | FAUX] Le paramètre « T » dans « template <class T> » est obligatoirement un type primitif défini par le langage « C++ », comme « int », ou « double », etc.

1.3 [VRAI | FAUX] En « C++ », une exception qui n'est pas capturée est ignorée par le programme.

1.4 [VRAI | FAUX] Cette instruction va générer une erreur à la compilation : typedef set<int> X;

1.5 [VRAI | FAUX] Il est possible d'utiliser la fonction « free » pour libérer l'espace mémoire alloué avec « new ».

1.6 [VRAI | FAUX] Il n'est pas possible de modifier la taille d'un tableau alloué dynamiquement avec « new ».

1.7 [VRAI | FAUX] La mémoire a été allouée pour un tableau avec l'appel « new[ ] » et cette même mémoire a été libérée avec l'appel « delete ». Le compilateur va rattraper votre erreur et générer une erreur de compilation pour non-concordance des appels.

1.8 [VRAI | FAUX] Le programme va planter si « new » n'arrive pas à allouer de la mémoire dans l'instruction suivante « Test* p = new Test(); », sachant que « Test » est une classe quelconque.

1.9 [VRAI | FAUX] Il est possible de déclarer une variable statique dans le fichier numéro -1- et l'utiliser dans le fichier numéro -2-?

1.10 [VRAI | FAUX] Il est possible de créer une situation de blocage (« deadlock ») même s'il n'y a qu'un seul programme en cours d'exécution et que ce programme ne se sert que d'un seul thread?

Exercice 2 (15 points)

2.1 Quelle est la différence entre une « frame » et une fenêtre (« window »).

2.2 Quand faut-il dériver les classes de « wxFrame » et quand faudrait-il les dériver de « wxWindow »?

2.3 Nous sommes en présence d'un programme très simple de « wxWidgets » (le premier exemple du cours). Nous avons décidé de retirer une ligne de ce programme. Le programme a compilé sans problème. Par contre lors de l'édition de liens, le compilateur nous a affiché ce message (sous Windows) :

```
[Linker Error] Undefined reference to « WinMain@16 »
ld returned 1 exit status
```

Quelle est cette fameuse ligne qui a provoqué cette erreur? Et pourquoi?

2.4 Expliquer pourquoi vous avez besoin de « sizer » pour disposer des boutons, mais vous n'en avez pas besoin pour disposer des menus.

2.5 Quelle est la différence entre un dialogue « modal » et un dialogue « non modal »?

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

```
01  #include <iostream>
02  using namespace std;
03
04  template<typename T> void Affiche(T x) { std::cout << x; }
05
06  class X {
07  public:
08      X() { Affiche('0');}
09      ~X() { Affiche(1);}
10  };
11
12  int main() {
13      try {
14          Affiche(5);
15          throw X();
16          Affiche(6);
17      } catch (X e) {
18          Affiche(7);
19      }
20      Affiche(8);
21      std::cout << std::endl;
22      return 0;
23  }
```

Tout en développant votre réponse, que va-t-il afficher en sortie le précédent programme?

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

```
01  #include <iostream>
02  #include <vector>
03  #include <iterator>
04  using namespace std;
05
06  void UneFonction(vector<int>& unvec) {
07        for (unsigned int i = 0; i < unvec.size(); i++) {
08              int n = unvec[i];
09              unvec.erase(unvec.begin()+i);
10              if (n % 2 == 0) {
11                    unvec.push_back(i);
12              }
13        }
14        copy(unvec.begin(),unvec.end(),ostream_iterator<int>(cout," "));
15        cout << endl;
16  }
17
18  int main(){
19        vector<int> v1 = {5, 2, 5, 2};
20        UneFonction(v1);
21
22        vector<int> v2 = {3, 5, 8, 9, 2};
23        UneFonction(v2);
24
25        vector<int> v3 = {0, 1, 4, 3, 1, 3};
26        UneFonction(v3);
27
28        return 0;
29  }
```

4.1 Tout en développant votre réponse, que va afficher en sortie l'appel de la ligne « **20** »?

4.2 Tout en développant votre réponse, que va afficher en sortie l'appel de la ligne « **23** »?

4.3 Tout en développant votre réponse, que va afficher en sortie l'appel de la ligne « **26** »?

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

```
01  #include <iostream>
02  #include <string>
03  #include <sstream>
04  #include <vector>
05  #include <utility>
06  #include <algorithm>
07  #include <iterator>
08
09  using namespace std;
10
11  string nuit(pair<int, int> &p){
12     string resultat;
13     resultat = to_string(p.first) + ":" + to_string(p.second);
14     return resultat;
15  }
16
17  int main(){
18     pair<int, int> p1(1,1);
19     pair<int, int> p2(2,2);
20     ostringstream oss;
21     vector<pair<int, int>> v;
22     v.push_back(p1);
23     v.push_back(p2);
24     auto res = nuit(p1);
25     cout << res << endl;
26     copy(v.begin(),v.end(),back_inserter(v));
27     cout << v.size() << endl;
28     transform(v.begin(),v.end(),ostream_iterator<string>(oss, ","), nuit);
29     auto valeur = oss.str();
30     cout << valeur.substr(4,3) << endl;
31
32     return 0;
33  }
```

5.1 Tout en expliquant votre réponse, quel est le type de la variable « **res** » de la ligne « **24** »?

5.2 Tout en expliquant votre réponse, que va afficher en sortie la ligne « **25** »?

5.3 Tout en expliquant votre réponse, que va afficher en sortie la ligne « **27** »?

5.4 Tout en expliquant votre réponse, que va afficher en sortie la ligne « **30** »?

Exercice 6 (20 points) la classe « Set_List » est un conteneur un peu spécial. Il permet de stocker des éléments dans un conteneur « List » tout en respectant les propriétés d'un conteneur « Set ». Ainsi, il n'est pas permis de préserver des valeurs dupliquées dans le conteneur « Set_List ».

```
1   class Set_List{
2   private:
3       list<string> s_list;
4   public:
5       Set_List() { }
6       void add(string);
7       bool remove(string);
8   };
```

```
01  int main(int argc, char * argv[]){
02      Set_List untest;
03      untest.add("bon");
04      untest.add("ete");
05      untest.add("bon"); // ignoré comme dans « Set »
06      untest.add("ete"); // ignoré comme dans « Set »
07      cout << untest.remove("bon") << endl; // affiche 1
08      cout << untest.remove("ete") << endl; // affiche 1
09      cout << untest.remove("bon") << endl; // affiche 0
10      cout << untest.remove("ete") << endl; // affiche 0
11
12      return 0;
13  }
```

Écrivez le code des méthodes « void add(string); » et « bool remove(string); ».

Lors de la correction, je vais tenir compte de la clarté de votre code et de son optimisation.

Joyeuses Fêtes et Bonne Année!