C++ PoP MT & EL /White vendredi 6 mai 2022

Ne PAS retourner ces feuilles avant d'en être autorisé!

Merci de poser votre carte CAMIPRO en évidence sur la table.

Vous pouvez déjà compléter et lire les informations ci-dessous:

NOM en MAJUSCULE _____

Prénom en MAJUSCULE _____

Numéro SCIPER _____

Signature _____

BROUILLON : Ecrivez aussi votre NOM-Prénom sur la feuille de brouillon fournie. Toutes vos réponses doivent être sur cette copie d'examen. Les feuilles de brouillon sont ramassées pour être immédiatement détruites.

Le test écrit commence à: **14h15**

Les copies d'examens sont ramassées à : **15h45**

Le contrôle de C++ PoP reste SANS appareil électronique

Vous avez le droit d'avoir tous vos documents **personnels** sous forme papier: dictionnaire, livres, cours, exercices, code, projet, notes manuscrites, etc...

Vous pouvez utiliser un crayon à papier et une gomme

Ce contrôle écrit de C++ PoP permet d'obtenir **35 points** sur un total de 100 points pour le cours complet.

1) (6 pts) Analyse de code

```
1   #include <iostream>
2
3   class Vector3D{
4   public:
5        Vector3D(const Vector3D &v){
6           x = v.x; y = v.y; z = v.z;
7        }
8        int getX() { return x; }
9        int getY() { return y; }
10       int getZ() { return z; }
11  private:
12       int x, y, z;
13  };
14
15  int main(){
16       Vector3D v1;
17       Vector3D v2(v1);
18       std::cout << "x = "   << v2.getX()
19                 << ", y = " << v2.getY()
20                 << ", z = " << v2.getZ();
21       return 0;
22  }
```

1.1) Choisir une réponse parmi les suivantes

Réponse	Description	Cocher une case
A	On obtient un exécutable qui affiche : **`x = 0, y = 0, z = 0`**	
B	On obtient un exécutable qui affiche des valeurs quelconques pour les attributs x, y et z	
C	L'exécution se termine anormalement, par exemple avec un message **`segmentation fault`**	
D	Une erreur est détectée à l'étape de la **compilation**	
E	Un erreur est détectée à l'étape de **l'édition de liens**	

1.2) **Justifier votre réponse** à la question précédente ; si vous avez choisi les réponses C à E alors précisez la nature de l'erreur et les instructions qu'il faut modifier ou ajouter pour supprimer le problème.

2) (9 pts) Surcharge d'opérateurs

Ce code compile sans warning en C++11 et s'exécute normalement

```
1   #include <iostream>
2   using namespace std;
3
4   class Complex
5   {
6   public:
7       Complex(): m_real(0.0), m_imag(0.0){ }
8       Complex(double real, double imag): m_real(real), m_imag(imag){ }
9       Complex(double real): m_real(real), m_imag(0.0){ }
10
11      friend Complex operator+(const Complex &c1, const Complex &c2);
12      friend ostream& operator<<(ostream& os,const Complex& c);
13
14  private:
15      double m_real;
16      double m_imag;
17  };
18
19  Complex operator+(const Complex &c1, const Complex &c2)
20  {
21      Complex c;
22      c.m_real = c1.m_real + c2.m_real;
23      c.m_imag = c1.m_imag + c2.m_imag;
24      return c;
25  }
26
27  ostream& operator<<(ostream& os, const Complex& c)
28  {
29      return os << c.m_real << " + " << c.m_imag << 'i' << endl;
30  }
31
32  int main()
33  {
34      Complex c1(1., 2.);
35      Complex c2(c1 + 3.4);
36      Complex c3(3.4 + c2);
37
38      cout << c1;
39      cout << c2;
40      cout << c3;
41
42      return 0;
43  }
```

2.1) indiquez le résultat de l'exécution de chacune des lignes de code suivantes et justifier brièvement comment la valeur de la variable affichée est initialisée :

Ligne 38 :

Justifier en détaillant l'initialisation de c1 à la ligne 34:

Ligne 39 :

Justifier en détaillant l'initialisation de c2 à la ligne 35:

Ligne 40 :
Justifier en détaillant l'initialisation de c3 à la ligne 36 :

2.2) Le mot-clef **friend** des lignes 11 et 12 est-il nécessaire ? Que se passe-t-il si on le supprime ?

La suite de l'exercice cherche à <u>modifier le code pour ne pas utiliser le mot clef friend.</u>
Le code de la fonction main() ne doit pas changer.

2.3) La surcharge de l'opérateur << peut-elle être effectuée avec une surcharge interne ? Selon votre réponse indiquez ci-dessous le code qu'il faut supprimer/écrire pour pouvoir surcharger l'opérateur << sans utiliser le mot clef **friend** et obtenir le même résultat d'exécution que pour la question 2.1.

2.4) La surcharge de l'opérateur + peut-elle être effectuée avec une surcharge interne dans ce programme ? Selon votre réponse indiquez ci-dessous le code qu'il faut supprimer/écrire pour pouvoir surcharger l'opérateur + sans utiliser le mot clef **friend** et obtenir le même résultat d'exécution que pour la question 2.1

3) (6 pts) Correction de code :

Ce code compile sans warning en C++11 mais produit une erreur à l'exécution.

```
1   #include <iostream>
2
3   class A
4   {
5   public:
6       A(): val(0), p(nullptr) {}
7       A(int v): val(v), p(new int(val)) {}
8       ~A() { delete p; }
9       void print () { if(p) std::cout << *p << std::endl; }
10  private:
11      int val;
12      int *p;
13  };
14
15  int main ()
16  {
17      A a1(10);
18      {
19          A a2;
20          a2 = a1;
21      }
22      a1.print();
23
24      return 0;
25  }
```

3.1) Pour quelle raison une erreur est-elle produite à l'exécution ?

Remarque : on ne demande pas le message d'erreur affiché par le système au moment de l'exécution mais seulement ce qui va causer un comportement incorrect à l'exécution.

3.2) Quelle méthode faut-il ajouter à la classe A pour que l'exécution ne produise plus de message d'erreur ? **On ne peut pas modifier la fonction main()** . Fournir sa définition complète ci-dessous :

4) (8 pts) Polymorphisme :

ce code compile sans warning avec l'option -std=c++11

```
1   #include <iostream>
2   using namespace std;
3
4   class BasicRobot {
5   public:
6        BasicRobot()  { f(); }
7        ~BasicRobot() { f(); }
8        void reboot() { f(); }
9   protected:
10       virtual void f() { cout << "Bip" << endl; }
11  };
12
13  class MobileRobot : public BasicRobot {
14  public:
15       MobileRobot() { f(); }
16       ~MobileRobot() { f(); }
17  protected:
18       virtual void f() { cout << "Vroum" << endl; }
19  };
20
21  class EmergencyRobot : public MobileRobot {
22  protected:
23       void f() { cout << "Pin-pon" << endl; }
24  };
25
26  int main(void) {
27       BasicRobot* pRobot = new EmergencyRobot;        //Q0
28       //pRobot->reboot();                             //Q1
29       //delete pRobot;                                //Q2
30       return 0;
31  }
```

Chaque question est liée à certaines lignes du code indiquées par un commentaire en fin de ligne.

Si l'instruction d'une question est commentée (Q1 et Q2) vous devez indiquer si le code s'exécute sans erreur quand on enlève le commentaire et, si oui, il faut indiquer le résultat de l'exécution. Si l'instruction ne s'exécute pas normalement la ligne reste en commentaire.

Question / Ligne(s)	Affichage obtenu ou description de la cause de l'erreur
Q0, ligne 27	

Q1, ligne 28	
Q2, ligne 29	

5) (6 pts) Héritage multiple :

5.1) Ce code produit une erreur de compilation.

```
1   #include <iostream>
2
3   class X {
4   public:
5        X() { b = 2; }
6        int b;
7   };
8
9   class Y : public X {
10  public:
11       Y() { b += 3; }
12  };
13
14  class Z : public X {
15  public:
16       Z() { b *= 4; }
17  };
18
19  class M : public Y, public Z {};
20
21  int main() {
22       M* m = new M();
23       std::cout << m->b << std::endl ;
24       return 0;
25  }
```

Préciser la nature de l'erreur de compilation détectée dans ce programme :

5.2) Corriger ce problème en conservant la hiérarchie de classes et sans modifier main().

5.3) Quel est l'affichage obtenu après la correction apportée par la question précédente ?
Justifier comment cet affichage est obtenu.