Fiche de TP2 : Notion de classe en C++

Pour gagner du temps dans la rédaction des bouts de programmes fournis dans les exercices, nous rappellons que les fichiers sources de ces programmes sont téléchargeables à l'adresse
http ://www.math.u-psud.fr/~apoung/mywepage/M2/codeTP2.tar.gz

Thème - *1 Apprentissage du cours*

Exercice-*1* **:**

- *Créer une classe qui contient un* **`int`** *(un entier).*
- *Implémenter l'* **`operator<<`** *pour l'afficher.*
- *Tester la solution.*

Exercice-*2* **:**

- *Créer une classe contenant un* **`int`**.
- *Ajouter une méthode pour l'afficher.*
- *Implémenter les opérateurs* **`++`** *préfixe et suffixe incémentant cet entier.*
- *Prouver que la solution fonctionne.*

Exercice-*3* **:** *On donne le programme suivant*

fichier : methodeprivees.cpp

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

classe Foo{
 private:
  Foo(){}
  ~Foo(){}
};

int main(int argc, char** argv)
{
  Foo C;
return 0;
}
```

- *Expliquer pourquoi ce programme ne compile pas.*
- *Donner deux façons de créer un objet de la classe* **`Foo`**, *sans changer de modificateur.*
  Une approche consistera à ajouter dans la classe **`Foo`** deux méthodes **`statiques`** créant et détruisant les objets de type **`Foo*`** ; une autre approche consistera à utiliser une **`classe amie`**, pour le faire.

Exercice-*4* **:** *On considère le programme suivant :*

fichier : temporaryobjects.cpp

```
#include <iostream>
using namespace std;
struct C{};
C f(){return C();}
void g(C& c){}
int main(int argc, char** argv)
```

```
{
  g(f());
return 0;
}
```

- *Ce programme compile-t-il ? Justifier.*

Exercice-*5* : *On considère le programme suivant :*

fichier : assignation.cpp

```
#include <iostream>
using namespace std;
class C{
  int id;
public:
  C(int i): id(i) {cout<<" C::C(int) : "<<id <<endl;}
  C& operator=(const C& c){
    id =c.id;
    cout<<"C::operator =(const C&) : "<<id<<endl;
  return *this;
  }
};

int main(int argc, char** argv)
{
  C c1(1);
  C c2 = c1;
return 0;
}
```

- *Qu'affiche-t-il ? Justifier.*
- *Il y a une erreur grave. Corriger.*

Exercice-*6* : *On considère le programme suivant :*

fichier : objectsplicing.cpp

```
#include <iostream>
using namespace std;
class C{
public:
  virtual void f(){cout <<"C::f()" << endl;}
};
class D: public C{
public:
  virtual void f(){cout <<"D::f()" << endl;}
};
void f(C c){
  c.f();
  D* d = reinterpret_cast<D*> (&c); //cast le plus violent; à éviter autant que possible
  (*d).f();
}
int main(int argc, char** argv)
{
  D d; f(d);
return 0;
}
```

- *Qu'affiche-t-il ? Expliquer.*

Thème - *2 Applications : une classe pour la gestion d'éléments de* $\mathbb{R}^2$

Exercice-*1* : *Écrire la classe* **`R2`**.

▲ *Utiliser un tableau de* 2 **`doubles`** *pour stocker* x *et* y

▲ *Coder les fonctions d'accès à* x *et* y *de deux manières :*

- *créer les fonctions*
  ```
  double& x()
  double& y()
  const double& x() const
  const double& y() const
  ```
- *ainsi qu'un accès par numéro de composante en surchargeant l'opérateur [].*

▲ *Programmer les opérations standards (somme, différence, ajout, retranchement, produit scalaire, produit vectoriel, produit par un scalaire, . . .).*

Démontrer le fonctionnement des fonctions écrites dans le programme principal.