Reda Touarga
Eliott BLANCHARD
SEC28

29/01/2026

Compte - rendu TP7_class
SMP

Sommaire

L’objectif de ce TP est de **manipuler les classes**, **surcharger des opérateurs** et **mettre en œuvre le polymorphisme en C++**.

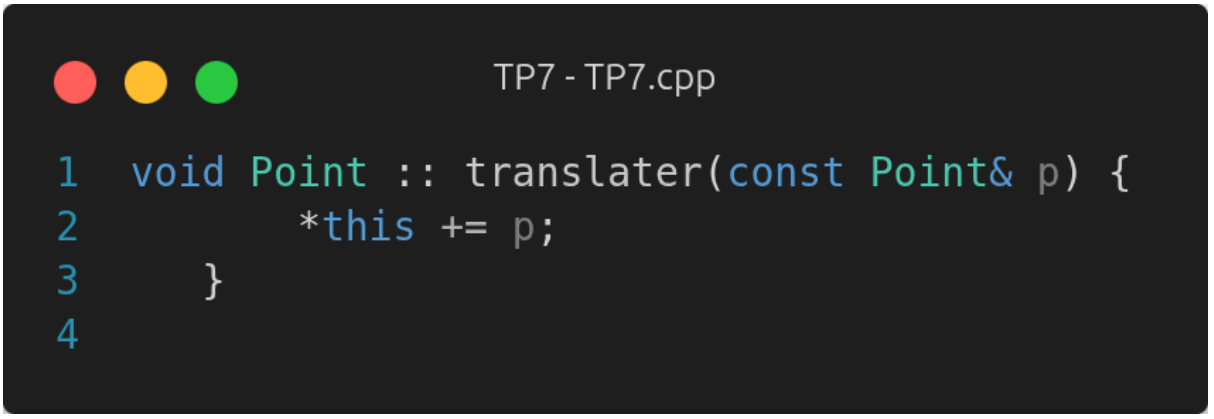
1. Création d’une classe Point

- Créer une classe Point (x, y réels)

TP7 - TP7.h

```
1  class Point {
2  private :
3      double x;
4      double y;
5  public :
6      //Constructeur
7      Point();
8      Point(double x, double y);
9      Point(Point const &p);
10     void Point :: translater(const Point& p);
11     //Accesseur
12     float get_x() const;
13     float get_y() const;
14     //Mutateur
15     void set_x(float x);
16     void set_y(float y);
17     Point& operator +=(const Point& p);
18 };
19
```

- Méthodes translater(Point) et translater(double, double)



- Constructeur par défaut, constructeur avec deux réels, constructeur de recopie

TP7 - TP7.cpp

```cpp
// Constructeur
Point :: Point() {
    x = 0.0;
    y = 0.0;
}

Point :: Point (double _x, double _y)   {
    x = _x;
    y = _y;
}

Point :: Point(Point const &p)  {

    x = p.x;
    y = p.y;

}
```

- Accesseurs et mutateurs

TP7 - TP7.cpp

```cpp
//Accesseur
float Point :: get_x() const{
    return x;
}

float Point :: get_y() const {
    return y;
}
// Mutateur

void  Point :: set_x(float x) {
    this->x = x;
}

    void  Point :: set_y(float y) {
    this->y = y;
}
```

2. Surcharge d’opérateurs

- Surcharger << pour afficher un Point

TP7 - TP7.cpp

```cpp
std::ostream& operator<<(std::ostream& cout, const Point& p) {
    cout << "(" << p.get_x() << ", " << p.get_y() << ")";
    return cout;
}
```

- Surcharger += pour translater un Point

TP7 - TP7.cpp

```cpp
Point& Point :: operator+=(const Point& p)   {
    this->x += p.x;
    this->y += p.y;
    return *this;

}
```

- Tests de validation

3. Formes géométriques abstraites

- Créer une classe abstraite Forme centrée sur un Point

TP7 - TP7.h

```
1  class Forme {
2  private :
3      Point centre;
4  public :
5      Forme(const Point& p);
6      Forme& operator +=(const Point& p);
7      Point get_centre() const;
8      virtual void perimetre() = 0;
9      virtual void surface() = 0;
10 };
11
```

- Surcharger += pour translater une forme

TP7 - TP7.cpp

```cpp
Forme& Forme :: operator+=(const Point& p) {
    this->centre+= p;
}
```

- Surcharger << pour afficher une forme

TP7 - TP7.cpp

```cpp
std::iostream& operator<<(std::iostream& cout, const Forme& f)  {
    cout << "(" << f.get_centre() << ")";
       return cout;
}

```

- Déclarer perimetre() et surface() abstraites

TP7 - TP7.h

```cpp
virtual void perimetre() = 0;
virtual void surface() = 0;
```

---

4. Formes géométriques concrètes

- Créer Cercle, Rectangle, Carré (Carré hérite de Rectangle)
- Côtés alignés avec les axes
- Implémenter perimetre() et surface()
- Surcharger << pour afficher type et attributs
- Tests + diagramme de classes