Programmation Objet

-

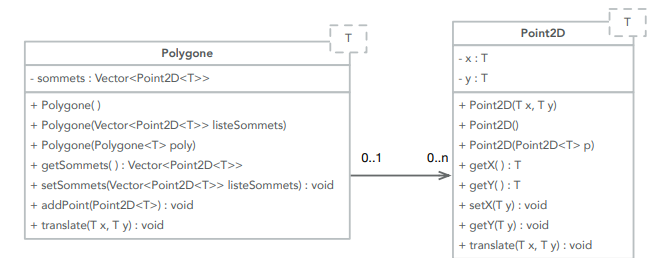
TP4 : Réalisation d’un cadastre de parcelles



# Objectif du TP

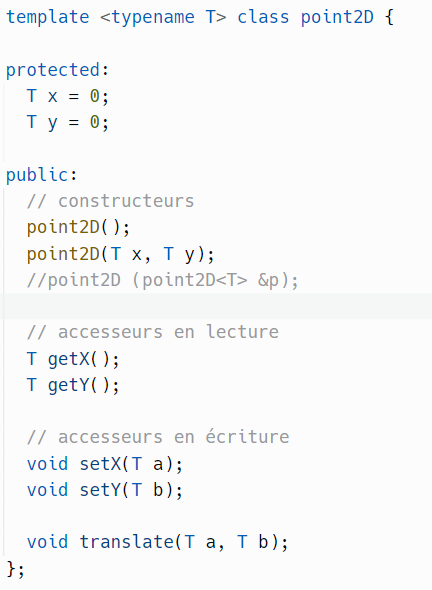
//A ecrire…

D’après les attendus définis dans le diagramme de classes suivant, nous avons codé les 2 classes Templates Point 2D et Polygone



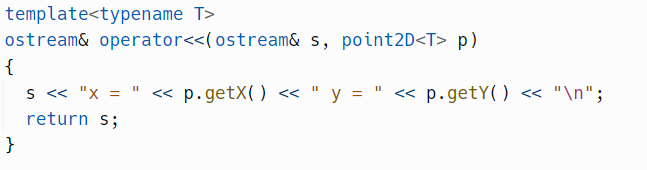
# Classe Point2D

Voici l’en-tête de la classe Point2D :



On y retrouve les attributs (les coordonnées du point), ainsi que les 8 méthodes qui apparaissent dans le diagramme de classes.

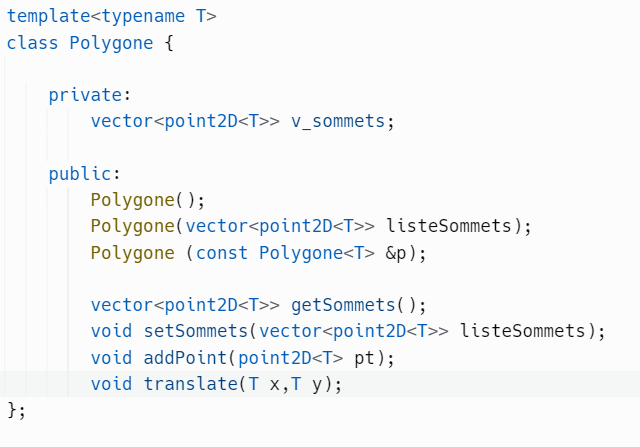
Voici le code permettant la surcharge de l’opérateur << pour l’affichage d’un point2D :



Vous pourrez retrouver le code de l’ensemble des méthodes de la classe point2D dans notre fichier point2D.h.

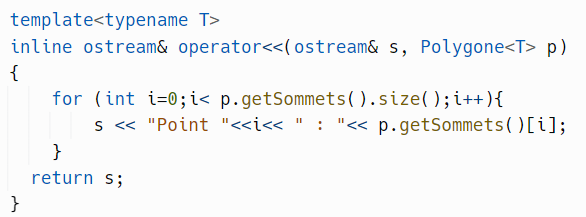
# Classe Polygone

Voici l’en-tête de la classe Polygone :



Vous retrouverez l’intégralité du code dans notre fichier Polygone.h.

Voici par exemple le code permettant la surcharge de l’opérateur << pour l’affichage d’un Polygone :



Voici un jeu d’essai avec la recopie de point2D et de Polygone, et la translation de Polygone :

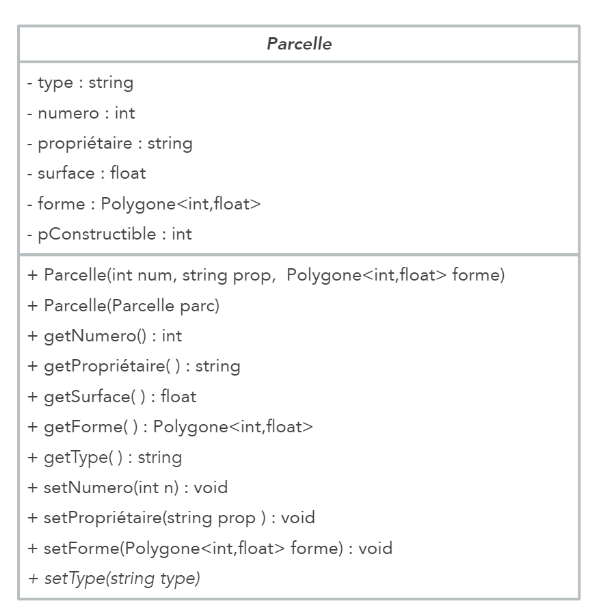


On remarque que le Polygone recopié n’est pas translaté après la translation du Polygone initial. C’est donc conforme à l’énoncé du TP.

# Parcelle

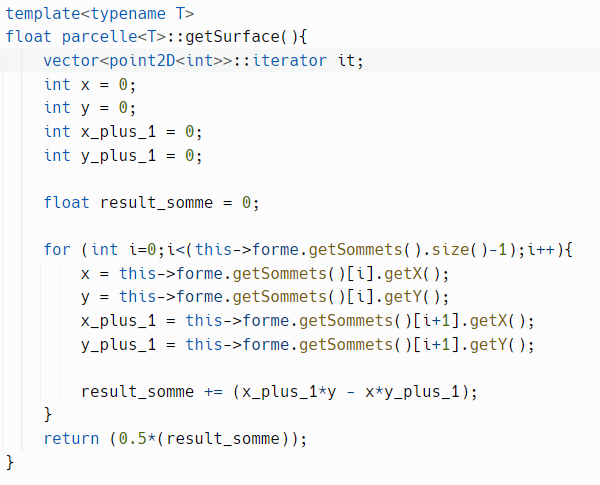
## Classe Parcelle

Voici le diagramme de classe de la classe Parcelle :



Nous avons donc écrit la classe Parcelle en remplissant tous ces critères. Seulement, nous avons décidé de la coder en tant que classe Template. Cela nous arrangeait pour la suite du TP.

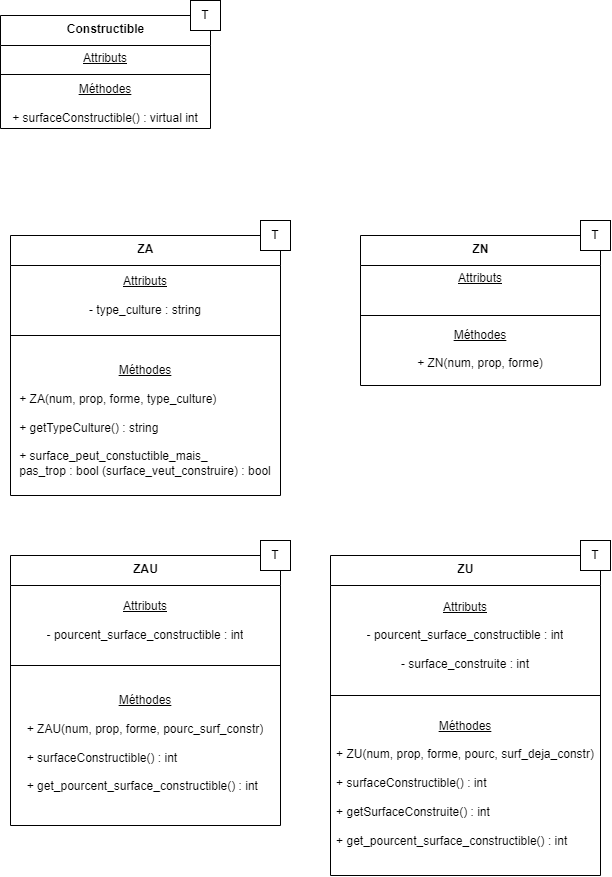
Voici par exemple la fonction permettant de calculer la surface d’une parcelle :

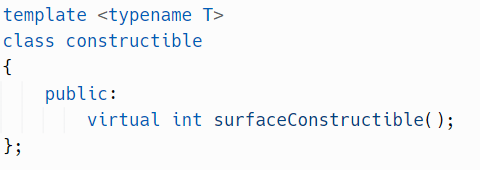


## Classe Constructible

Nous avons eu du mal à intégrer la classe Constructible à notre code.

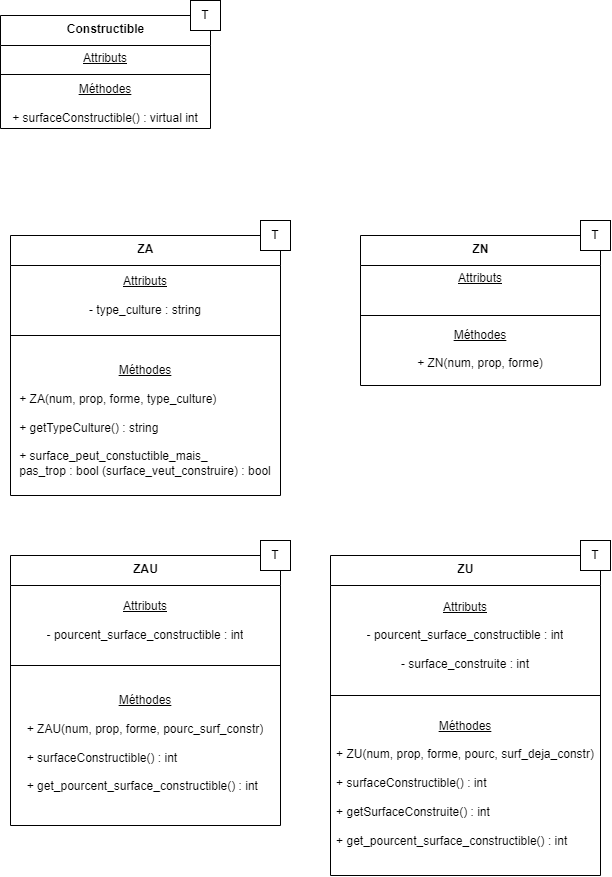
Nous avons finalement codé cette classe pour utiliser le concept d’héritage multiple, sans attributs et avec une méthode en virtual, mais elle n’a pas d’intérêt particulier dans notre code. Voici son diagramme de classe ainsi que son en-tête :





## Classes héritées des classes Parcelle et Constructible

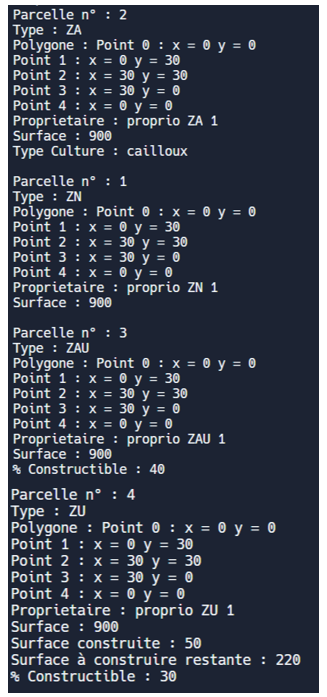
Nous avons ensuite décidé de créer 4 « sous-classes » qui représentent les différents types de parcelles. Elles héritent toutes les 4 des classes Constructible et Parcelle, et ont des attributs et des méthodes propres à leur description, comme illustré sur ce diagramme de classes :



De même que pour la classe Constructible, ces 4 classes ont été définies en tant que classes Template.

## Affichage des Parcelles

Voici l’affichage de 4 Parcelles de classes différentes, que l’on obtient grâce à une surcharge de l’opérateur d’affichage dans chacune des 4 classes. On note qu’il est identique à celui proposé dans l’énoncé :

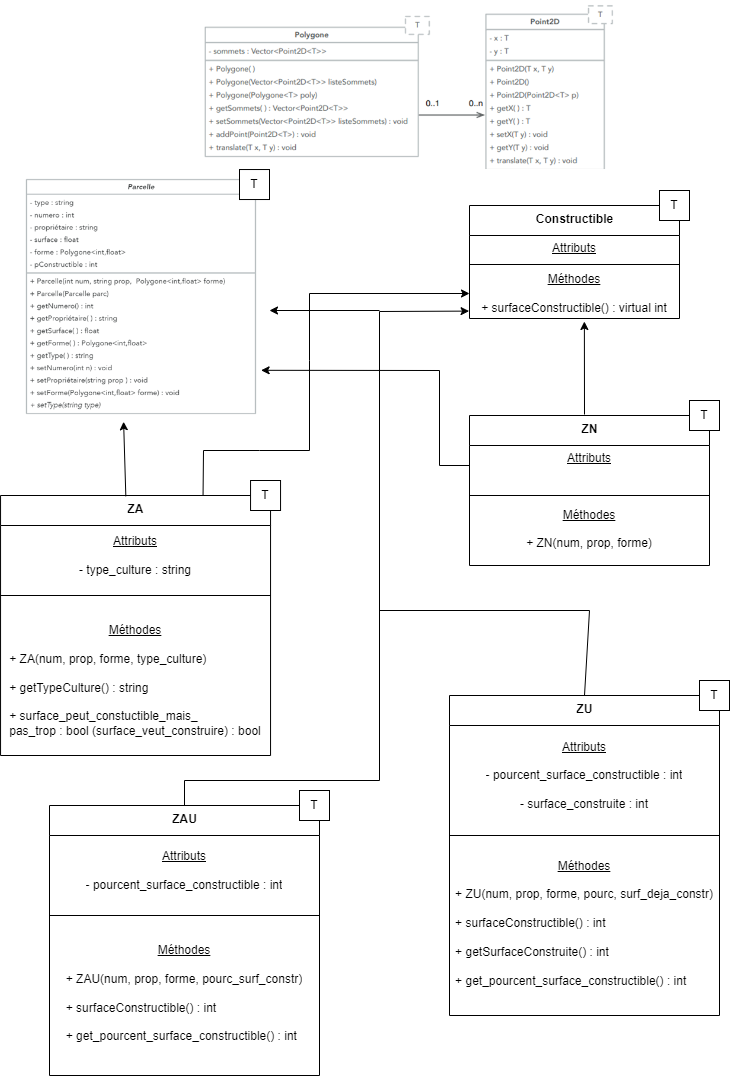


# Carte

//J’ai pas tout compris dans ton code Robin et j’ai pas eu le temps d’approfondir…

# Diagramme de classes finale du TP

Voici le diagramme de classes de notre code, qui a été modifié tout au long du projet.



//Je vous ai envoyé par mail centrale le diagramme sur draw.io, il n’y a plus qu’à ajouter les classes de la carte

# Conclusion