Templates
Objectif 1 - Devant ou derrière ?

2. On va faire un template, car notre booléen est constant et on est égale à false donc
on fait un template sur la fonction add_waypoint avec un booléen constant en paramètres et
on appel ad__waypoints avec en faisant add_waypoints<false> car le front était constant et
égale à false.

Objectif 2 - Points génériques

1. On va faire une struct avec un template qui prend une taille et un type pour les points,
template<const int taille,typename type>, type sera entier, float et double.

3. J'ai choisi de faire un alias dans la classe point que j'appelle self_type et qui reprsente un
Point avec la taille et le type que j'ai mis. On doit faire en sorte de mettre toutes les fonctions
de Point2D et Point3D dans la classe pour que tout compile en remplacant les Point2D et Point3D par
le self_type que j'ai crée. Puis en dehors de la struct on va faire les 2 alias Point2D et Point3D
en les appelant avec leurs tailles et leurs type donc float. Sans oublier de faire les différents
constructeurs du Point2D Point3D.

4. Les erreurs se produisent que maintenant car on a des problèmes de types entre les différents
opérateurs maintenant après avoir tout implémenté.

5. On va faire plusieurs conditions dans les 2 constructeurs avec le static_assert et aussi dans
les fonctions y() et z(), le but est de verifier si la taille donnée au templates correspond bien
au nombre d'arguments. Donc on va vérifier pour le constructeur du Point2D si son Size est bien de 2,
et pour le constructeur de Point3D c'est pareil mais avec 3. Pour y() on doit vérifier si la taille est
plus grand ou égale à 2 car il faut vérifier si on est minimum en 2D et pour z() c'est pareil avec 3 pour
vérifier si on est en minimum 3D.