Assertions et exceptions

Objectif 1 - Crash des avions

1. Pour eviter que le code crash on va utiliser un try catch pour catch l'erreur et eviter qu'elle se propage
jusqu'à la fin du programme et fais tout crasher. Donc on fais un try catch sur le move de Aircraft qu'on
appel dans le move de AircraftManager, on catch la AircraftCrash erreur et affiche le message sur cerr avec
what().

2. On a juste à créer un champs m qui sera privé dans AircraftManager et qui va etre incrémenté à chaque
catch, on va pouvoir l'appeler avec une méthode publique, qui sera un getter pour récupérer la variable.
Puis on ajoutera une touche m dans notre fonction create_keystrokes de tower_sim qui affichera un petit
message avec le nombre d'avion qui a crash (donc m).

3. On va lancer une exception qui sera l'AircraftCatch et on met un message dedans (avec les { msg }), qui
va lancer cette exception et affichera message si on appel what() sur l'erreur. Donc elle sera aussi affiché à l'appel du try/catch dans move de AircraftManager.

Objectif 2 - Détecter les erreurs de programmation

Struct disp_z_cmp ->
bool operator()(const Displayable* a, const Displayable* b) const :
    assert(a != nullptr && b != nullptr); pour eviter que a et b soit null;

AircraftFactory -> void AircraftFactory::add_name(const std::string& name) :
    assert(name.empty()); // evite que le nom de l'avion est vide ce qui ne devrait pas arriver

AicraftManager -> Toutes les fonctions ou lambda qui ont en paramètres des unique_ptr d'Aircraft :
    assert(aircraft != nullptr); // evite qu'un avion soit null (le faire pour tout les avions utilisés)

Airport -> WaypointQueue start_path(const size_t terminal_number)
-> Terminal& get_terminal(const size_t terminal_num) :
    assert(terminal_num <= 2); // si le terminal est bien entre 0 (c'est un size_t impossible qu'il soit
    négatif) et 2 car il y a que 3 terminaux

AirportType -> WaypointQueue terminal_to_air(const Point3D& offset, const size_t runway_num,
                                                const size_t terminal_num) const
-> WaypointQueue air_to_terminal(const Point3D& offset, const size_t runway_num,
    assert(terminal_num <= 2); // si le terminal est bien entre 0 (c'est un size_t impossible qu'il soit
    négatif) et 2 car il y a que 3 terminaux

TowerSimulation -> void TowerSimulation::launch() :
        assert(airport != nullptr); // on verifie si l'aéroport est null après ca création au cas ou