Laboratorio de programación 1
Trabajo Práctico Nº2

1. Modelizar el UML de las clases pedidas con sus relaciones.
2. Generar los archivos pedidos de las clases que considere necesarias e implementar un main con las características mencionadas al final del archivo.
3. Generar los métodos to_string e Imprimir en pantalla para cada clase, ninguna debe recibir nada por parámetro y devuelve los atributos concatenados como un string
4. **¡¡Generar los getters y setters que crea necesarios!!**

Aeroparque los convoca para que colaboren con la implementación de un nuevo sistema de control de vuelos. Les piden que desarrollen un sistema para el control y cálculo de cargas en cada vuelo.

1) Un Pasajero que tiene un DNI, su nombre, la fecha, número de vuelo, asiento, lista de valijas que transporta (y el peso de cada una). Además, el método AgregarEquipaje que chequee que el equipaje total no supere los 25 kg por persona.
2) Cada Vuelo que contenga número de vuelo, estado, avión, si es partida o arribo, fecha y hora de partida del vuelo que toma, fecha y hora de arribo, aeropuerto de destino y los pasajeros y equipaje a transportar. El número de vuelo se debe generar automáticamente a medida que se crean objetos de la clase vuelo. Como método de la clase Vuelo deberá poder pasarle un DNI y obtener los datos del pasajero correspondiente y si tienen equipaje y su peso.
   Recomendación: Generar una lista global de la clase de strings que tenga los posibles destinos.
3) Un Avión que tiene un ID único, cantidad de pasajeros que permite, cantidad actual, peso máximo que puede transportar en **total**. El avión puede despegar, aterrizar, pedir permiso y recibir permiso tanto para aterrizar como para despegar. Realizando los siguientes chequeos para el despegue:
   a) El avión puede cargar cierto peso máximo, para calcular el peso del avión se considera la suma del peso humano y el peso total del equipaje, siendo el peso humano el peso de los pasajeros. Un pasajero tiene un peso promedio de 75kg (se considera lo mismo para todos los pasajeros) y el peso total del equipaje incluye todos los equipajes de todos los pasajeros. En este cálculo hay que considerar el peso de los 4 integrantes de la tripulación (no llevan equipaje).
   b) El método ChequearCargaMaxima debe verificar que el peso total sea menor que la capacidad máxima y lanza una excepción si esto no se cumple.
4) El Aeropuerto que tenga ID único, capacidad del aeropuerto (para almacenar aviones), lista de vuelos y una lista de los aviones que se encuentran actualmente en el aeropuerto. Los aviones se agregan a la lista cuando aterrizan y se quitan cuando despegan. El Aeropuerto puede DarPermiso que otorgue o niegue el permiso de aterrizar según la disponibilidad del hangar.
   i) El método tiene que lanzar una excepción en el caso de que el hangar esté lleno.
5) Implementar métodos que permitan:
   a) Antes de salir del Aeropuerto, el avión debe verificar que se cumplan las condiciones de vuelo (capacidad del avión y control de peso de equipaje).

b) Hacer método AgregarPasajero, CambiarPasajero o EliminarPasajero en la clase Vuelo.
c) Implementar métodos que calculen la cantidad de pasajeros que volaron en un día, la cantidad de vuelos que aterrizaron y despegaron en el día y el porcentaje de vuelos que despegaron y aterrizaron en horario.
d) Imprimir

6) Utilice como mínimo un atributo static, un atributo const y una clase friend.

Sobrecargar el operador de impresión para poder imprimir los datos de un vuelo.
Sobrecargar el método suma para poder agregar equipaje a un pasajero.
Sobrecargar el método resta para poder eliminar equipaje a un pasajero.
Sobrecargar el operador corchetes para acceder a un elemento de, al menos, una lista.

Generar el main que deberá simular el funcionamiento de Aeroparque (un solo aeropuerto). Pruebe todos los métodos. En todas las clases, puede agregar los atributos y métodos que considere oportunos.

IMPORTANTE, Esta consigna es la base de lo que necesitan hacer, cualquier cosa extra es bienvenida o si les interesa agregar algún concepto que hayan visto en algún lado pueden hacerlo, en caso que hagan eso no será considerado para la corrección, pero sí como concepto. En caso de que haya problemas con lo que le hayan querido agregar (Por ejemplo una librería gráfica, de sonido, etc) tendrán prioridad los que tengan problemas con la consigna original.

¡LOS MÉTODOS NO DEBEN RECIBIR NI ESCRIBIR EN CONSOLA! (Salvo método imprimir y si se aclara específicamente).

Genere el main y utilice todos sus métodos para verificar su funcionamiento creando todas como objetos dinámicos (Recuerde liberar la memoria al creer necesario).