Lab de Progra 3

Semana 1

Title: Fleet Operations Simulator/ Simulador de una Flota de Vehiculos

- Model a class **hierarchy** with **inheritance** and **polymorphism**.
- Use **raw pointers** safely with virtual destructors.
- Apply **dynamic_cast** to access an interface (**IFlyable**) from base pointers.
- Write a time-step simulation with loops and conditional actions.
- Produce a readable main that ties it all together.

Deliverable

- Create files (.h and .cpp) for all Base and Derived classes
- Console output showing the 8-hour simulation and final summary.

Fleet Simulator

Create a small simulator with a polymorphic base class **Vehicle**, and four vehicles:

- Car
- Truck
- Drone (also implements **IFlyable**)
- Motorcycle

The interface IFlyable for anything that can fly.

Store **raw pointers** (Vehicle*) in a std::vector, simulate 8 hours, attempt flights every 2 hours, refuel every 3 hours, and print a final fuel summary. Delete all pointers at the end.

```
struct IFlyable {
    virtual void fly(int minutes) = 0;
    virtual ~IFlyable() = default;
};
```

Base class Vehicle

Implement a polymorphic base class that every vehicle will derive from.

Requirements

- Fields: **string** name_; int fuelLevel_; (0–100).
- normalizeFuelLevel() keeps fuelLevel_ within [0,100].
- Ctor: Vehicle(**string** name, int fuelLevel = 100).
- **Virtual destructor**.
- Accessors: getName(), getFuelLevel().
- **Pure virtuals**: refuel(int), status() const.
- Virtual simulateHour() with a **default** 5-unit fuel burn and a **low fuel warning** (<20).

Hints

- Use std::max / std::min inside normalizeFuelLevel().

Derived classes

Create **four** derived classes with distinct behavior. Each must:

- Override refuel(int units), status() const, simulateHour().
    - Default refuel 15
- Call normalizeFuelLevel() after any change to fuel.

a) Car

- Hourly burn: ~6 units.
- refuel(units): add fully (+units).

b) Truck

- Hourly burn: ~9 units.
- refuel(units): less efficient (+units/2).

c) Drone (implements IFlyable)

- Hourly burn: ~8 units.

- refuel(units): charges fast (+(units*3)/2).
- fly(minutes): burns minutes*3. Flies 5 mins by default.
- Print a line when it flies.

d) Motorcycle

- Hourly burn: ~4 units.
- refuel(units): efficient but not instant; +(units*4)/5.
- Status should identify it as a Motorcycle.

In any overridden simulateHour(), print a specific low-fuel message for that type.

Expected output:

[0h]

Car: Toyota Camry 2010 | Fuel Level=94

Drone: DJI Mini 3 | Battery=92

[1h]

Car: Toyota Camry 2010 | Fuel Level=88

Drone: DJI Mini 3 | Battery=84

[2h]

Car: Toyota Camry 2010 | Fuel Level=82

Drone: DJI Mini 3 | Battery=76

Attempting flight for flyable vehicles...

DJI Mini 3 flew for 5 minutes.

[3h]

Car: Toyota Camry 2010 | Fuel Level=76

Drone: DJI Mini 3 | Battery=53

Refueling all vehicles (+15)

[4h]

Car: Toyota Camry 2010 | Fuel Level=85

Drone: DJI Mini 3 | Battery=67

Attempting flight for flyable vehicles...

DJI Mini 3 flew for 5 minutes.

[5h]

Car: Toyota Camry 2010 | Fuel Level=79

Drone: DJI Mini 3 | Battery=44

[6h]

Car: Toyota Camry 2010 | Fuel Level=73

Drone: DJI Mini 3 | Battery=36

Attempting flight for flyable vehicles...

DJI Mini 3 flew for 5 minutes.

Refueling all vehicles (+15)

[7h]

Car: Toyota Camry 2010 | Fuel Level=82

Drone: DJI Mini 3 | Battery=35

Final summary:

- Toyota Camry 2010 -> Fuel Level: 82

- DJI Mini 3 -> Fuel Level: 35