Vehicle Rental System – C++ Project

# 📌 Description

This project is a simple vehicle rental management system written in C++. It demonstrates object-oriented programming concepts like inheritance, polymorphism, and dynamic memory management.  
  
It allows different types of vehicles (Cars and Trucks) to calculate rental fees polymorphically over a given date range using pointer arithmetic and dynamic arrays.

# 🧱 Key Components

## 1. Date Struct

Stores rental start and end dates:

struct Date {  
 int day, month, year;  
};

## 2. Vehicle Abstract Base Class

Defines the common interface for all vehicle types:

class Vehicle {  
public:  
 virtual float calcRent(const Date\* start, const Date\* end) = 0;  
 virtual ~Vehicle() {}  
};

## 3. Car Class

Calculates rent as dailyRate × number\_of\_days:

class Car : public Vehicle {  
 float dailyRate;  
public:  
 Car(float rate);  
 float calcRent(const Date\*, const Date\*) override;  
};

## 4. Truck Class

Calculates rent as flatRate + (payloadSurcharge × number\_of\_days):

class Truck : public Vehicle {  
 float flatRate, payloadSurcharge;  
public:  
 Truck(float rate, float surcharge);  
 float calcRent(const Date\*, const Date\*) override;  
};

## 5. Pointer Arithmetic for Date Calculations

A helper function computes the number of days between two Date\* pointers using simplified logic (30 days per month, 360 days per year).

int computeDays(const Date\* start, const Date\* end);

# 🛠️ Functionality

## Add a Vehicle

void addVehicle(Vehicle\*\*& inventory, int& size, Vehicle\* v);

## Remove a Vehicle

void removeVehicle(Vehicle\*\*& inventory, int& size, int index);

# 🔁 Polymorphism in Action

Each vehicle in the Vehicle\*\* inventory array calls its own calcRent() method through polymorphic dispatch:

float rent = inventory[i]->calcRent(startDate, endDate);

# 📌 Example Output

Vehicle 0 rent: $200  
Vehicle 1 rent: $225  
  
After removing vehicle 0:  
Vehicle 0 rent: $225