**Car Functionality System**

# 1. Project Overview

This project simulates a basic car's functionality using Object-Oriented Programming in C++. It includes operations such as accelerating, braking, shifting gears, driving toward a destination, and checking if the car has arrived. The system tracks the car's location, calculates the remaining distance and estimated time to the destination, and is thoroughly tested using unit tests.

# 2. System Features

* Gear shifting: PARK, REVERSE, NEUTRAL, DRIVE
* Acceleration and braking
* Set and track current location and destination
* Drive the car based on speed and time
* Estimate time left to reach destination
* Check if the car has arrived at the destination
* Unit tested with assertions

# 3. Technical Specifications

## 3.1 GearType Enum

Defines possible gear positions:  
- PARK  
- REVERSE  
- NEUTRAL  
- DRIVE

## 3.2 Location Struct

Represents geographical coordinates:  
- latitude (double)  
- longitude (double)

## 3.3 Car Class

Encapsulates car behavior and state.

Private Members:

* GearType gearState
* double speed
* Location currentLocation
* Location destination

Public Methods:

* Car() – Constructor
* void accelerate(double value)
* void brake(double value)
* void shiftGear(GearType newGear)
* GearType getGearState() const
* double getSpeed() const
* Location getLocation() const
* Location getDestination() const
* void setDestination(const Location& dest)
* void drive(double timeInHours)
* double timeToDestination() const
* bool hasArrived() const

# 4. User Guide

How to use the Car class:

1. Create a Car object.
2. Set the destination using setDestination().
3. Shift the gear to DRIVE or REVERSE using shiftGear().
4. Accelerate the car using accelerate().
5. Call drive() with the time duration in hours to move the car.
6. Check if the car has arrived using hasArrived().
7. Get the estimated time left to destination using timeToDestination().

# 5. Build and Test Instructions

Requirements:  
- C++ compiler with C++17 support  
- CMake build system  
  
Steps:  
1. Clone the repository.  
2. Run 'cmake .' to configure the build.  
3. Run 'make' to build the project.  
4. Run './test\_car' to execute unit tests.

# 6. Test Cases

| **Test Case Name** | **Description** | **Input** | **Expected Output** |
| --- | --- | --- | --- |
| testInitialState | Verify initial car state | Create new Car object | Speed = 0, Gear = PARK, Location = (0,0) |
| testSetAndGetDestination | Set and get destination | Set destination (10, 20) | Get destination returns (10, 20) |
| testAccelerateAndBrake | Accelerate and brake functionality | Accelerate 50, Brake 20, Brake 100 | Speed after operations: 0 |
| testGearShift | Change gear states | Shift through PARK, DRIVE, etc. | Gear state matches shifted gear |
| testDriveAndArrival | Drive for given time and check arrival | Set destination (5,0), speed=10 | Arrival at destination after driving time |
| testTimeToDestination | Check time to destination calculation | Set destination, speed | Correct time returned |
| testSameLocationArrival | Arrival detection when at destination | Set destination = current location | hasArrived() returns true |

# 7. Notes

• The system approximates the arrival within 50 meters.  
• Speed is measured in units per hour (could be km/h).  
• Acceleration and braking values must be positive to affect speed.  
• Accelerating in gears other than DRIVE or REVERSE does not increase speed.