```cpp
#include "Car.h"
#include<iostream>
int main() {
    // Lecture 46: Const Member Functions
    const Car c(5);
    // if we declare the above car as const, we only can access methods which do not modify the state of the object.
    // These functions below - will all modify the state of the object, so all these accelerate calls which modify the speed and fuel will give errors.
    //car.Accelerate(); - so all these will result in errors.
    //car.Accelerate();
    //car.Accelerate();
    // However c.Dashboard does not modify the state of the object, so we should allow it to be qualified by const - so that this declaration will compile.
    c.Dashboard();

    // in car.cpp, where the c.Dashboard is implemented, do the following:
    // add the const at the back of the definition
    void Car:: Dashboard() const {
        fuel++;  // is under the hood actually impleemented as:
        this->fuel++;
        // if we modify the state of the object in a const qualified function, this-> indicates the object. The const qualifier that we have applied to the member function is actually ALSO applied to this pointer. More specifically, it is applied to *this, therefore when we try to modify the value of a member variable inside this member function it is not allowed. This also makes all of its members constant.

        // So the statement
        std::cout << fuel << std::endl;
```

```
            std::cout << speed << std::endl;
        }
        // in car.h.
        // add the const at the back of the declaration
        void Dashboard() const;


        return 0;
    }

```