```cpp
#include <iostream>
using namespace std;

class Car
{
protected:
    float fRadian;
    float fSpeed;
    float fDeltaTime;
public:
    void drive(float fRadian, float fSpeed, float fDeltaTime) {
        cout << "Driving..." << endl;
        cout << "Radian: " << fRadian << endl;
        cout << "Speed: " << fSpeed << endl;
        cout << "DeltaTime: " << fDeltaTime << endl;
    }
};

class AutopilotCarOne : public Car {
public:
    void autoDrive() {
        fRadian = rand() % 100;
        fSpeed = rand() % 100;
        fDeltaTime = rand() % 10;
        cout << "autoDriving..." << endl;
        drive(fRadian, fSpeed, fDeltaTime);
    }
};

class AutopilotCarTwo : protected AutopilotCarOne {
public:
    void autoDrive() {
        AutopilotCarOne::autoDrive();
    }
    void optimizedDrive(float fRadian, float fSpeed, float fDeltaTime) {
        fRadian += rand() / double(RAND_MAX);
        fSpeed += rand() / double(RAND_MAX);
        fDeltaTime -= rand() / double(RAND_MAX);
        drive(fRadian, fSpeed, fDeltaTime);
    }
};

class AutopioletCarThree : private AutopilotCarTwo {
public:
```

```
    void autoDrive() {
        AutopilotCarTwo::autoDrive();
    }
};

class UpgradedAutopilotCar : public AutopioletCarThree {
public:
    void driveWithMusic() {
        cout << "music" << endl;
        autoDrive();
    }
};

int main()
{
    Car car;
    car.drive(30.0, 60.5, 2.0);
    AutopilotCarOne car_one;
    car_one.drive(30.0, 60.5, 2.0);
    car_one.autoDrive();
    AutopilotCarTwo car_two;
    car_two.optimizedDrive(30.0, 60.5, 2.0);
    car_two.autoDrive();
    AutopioletCarThree car_three;
    car_three.autoDrive();
    UpgradedAutopilotCar music_car;
    music_car.autoDrive();
    music_car.driveWithMusic();
    return 0;
}
```