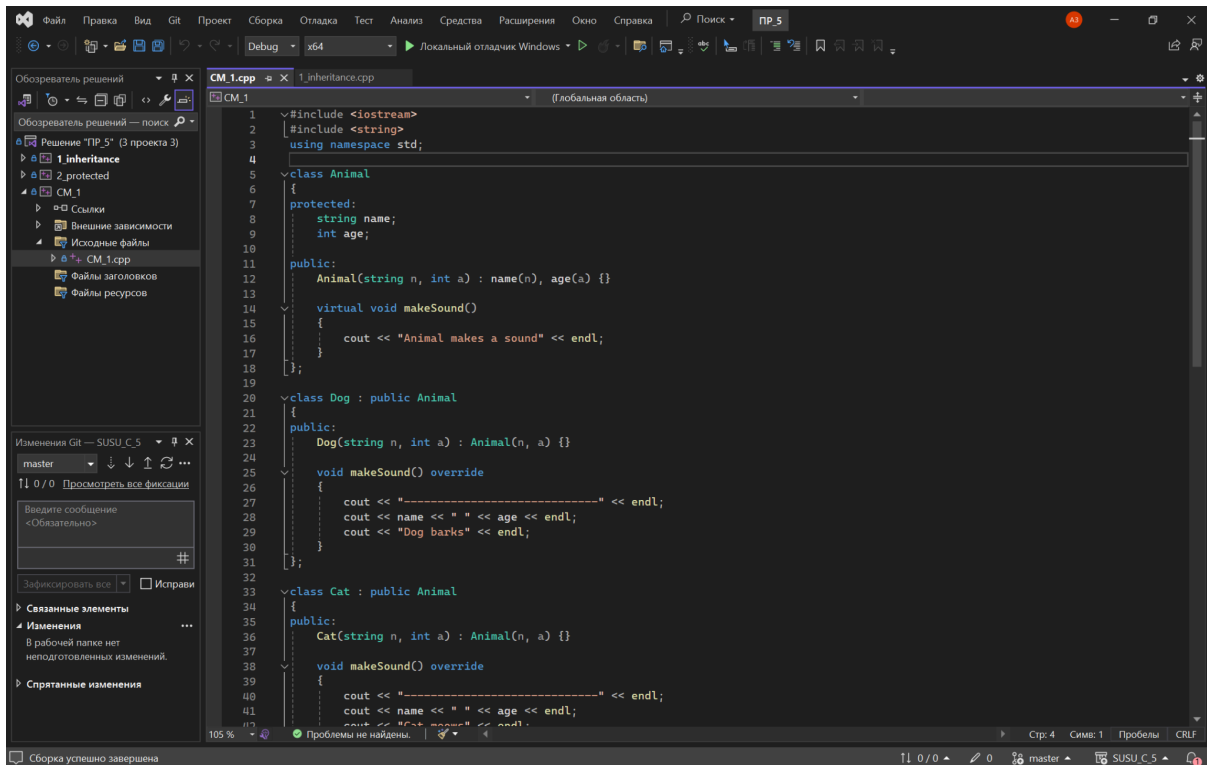
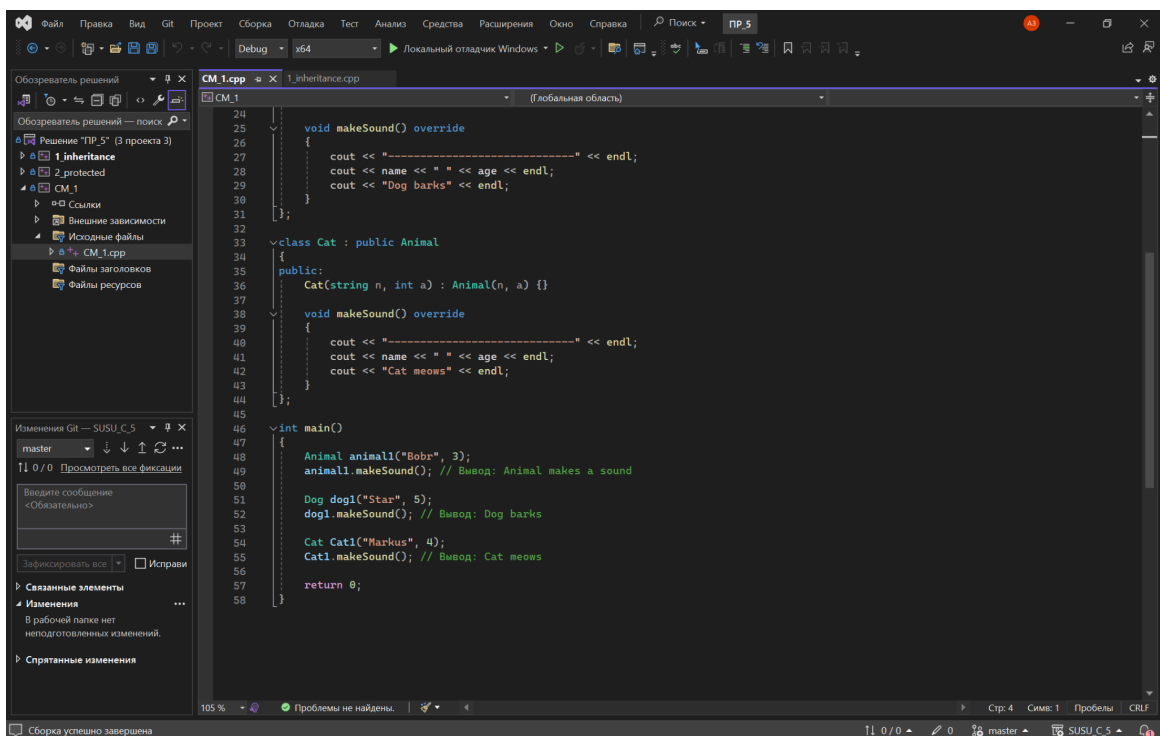


Самостоятельная работа



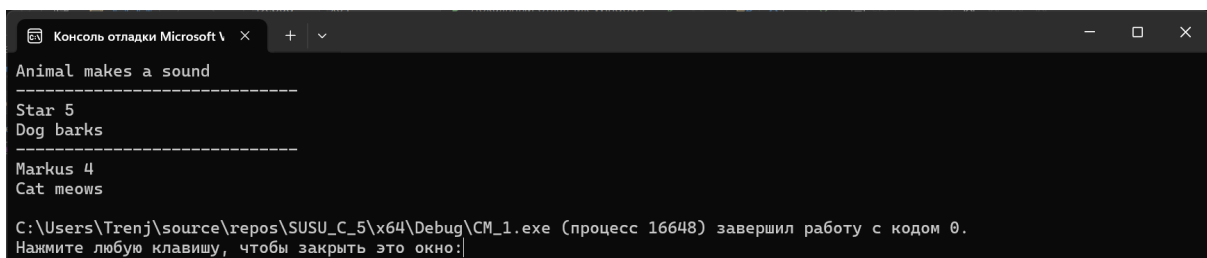
The screenshot shows the Visual Studio IDE with the file `1_inheritance.cpp` open. The code defines three classes: `Animal`, `Dog`, and `Cat`. `Animal` is the base class with a `protected` `name` and `age` attribute, and a `public` `makeSound()` method. `Dog` and `Cat` are derived from `Animal` and override the `makeSound()` method. The `main()` function is not yet implemented in this view.

```
1 #include <iostream>
2 #include <string>
3 using namespace std;
4
5 class Animal
6 {
7     protected:
8         string name;
9         int age;
10
11     public:
12         Animal(string n, int a) : name(n), age(a) {}
13
14         virtual void makeSound()
15         {
16             cout << "Animal makes a sound" << endl;
17         }
18 };
19
20 class Dog : public Animal
21 {
22     public:
23         Dog(string n, int a) : Animal(n, a) {}
24
25         void makeSound() override
26         {
27             cout << "-----" << endl;
28             cout << name << " " << age << endl;
29             cout << "Dog barks" << endl;
30         }
31 };
32
33 class Cat : public Animal
34 {
35     public:
36         Cat(string n, int a) : Animal(n, a) {}
37
38         void makeSound() override
39         {
40             cout << "-----" << endl;
41             cout << name << " " << age << endl;
42             cout << "Cat meows" << endl;
43         }
44 };
45
46
47
48
49
50
51
52
53
54
55
56
57
58
```



This screenshot shows the same Visual Studio IDE, but now the `main()` function is implemented. It creates an `Animal` object named `animal1` with name "Bobr" and age 3, a `Dog` object named `dog1` with name "Star" and age 5, and a `Cat` object named `Cat1` with name "Markus" and age 4. Each object's `makeSound()` method is called, resulting in the output shown in the console below.

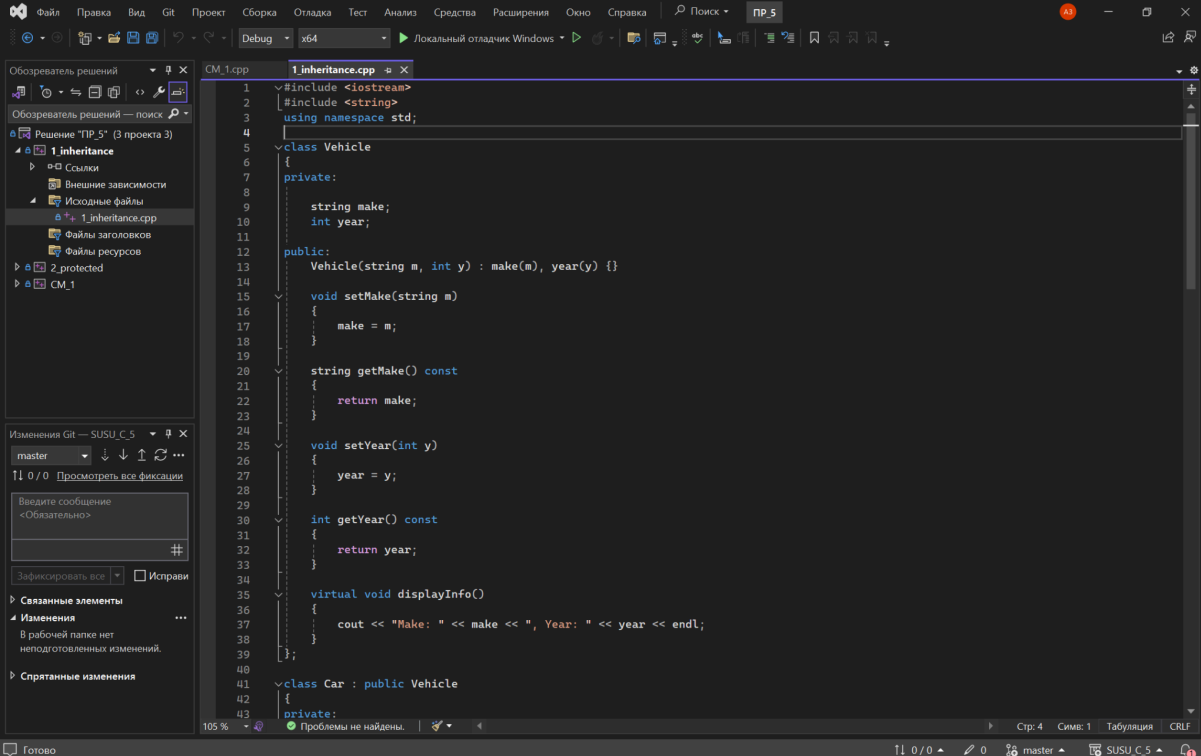
```
24
25     void makeSound() override
26     {
27         cout << "-----" << endl;
28         cout << name << " " << age << endl;
29         cout << "Dog barks" << endl;
30     }
31 };
32
33 class Cat : public Animal
34 {
35     public:
36         Cat(string n, int a) : Animal(n, a) {}
37
38         void makeSound() override
39         {
40             cout << "-----" << endl;
41             cout << name << " " << age << endl;
42             cout << "Cat meows" << endl;
43         }
44 };
45
46
47 int main()
48 {
49     Animal animal1("Bobr", 3);
50     animal1.makeSound(); // Вывод: Animal makes a sound
51
52     Dog dog1("Star", 5);
53     dog1.makeSound(); // Вывод: Dog barks
54
55     Cat Cat1("Markus", 4);
56     Cat1.makeSound(); // Вывод: Cat meows
57
58     return 0;
59 }
```



The screenshot shows the Microsoft Visual Studio Console window. It displays the output of the program: `Animal makes a sound`, `Star 5`, `Dog barks`, `Markus 4`, and `Cat meows`. Below the output, a message states: `C:\Users\Trenj\source\repos\SUSU_C_5\x64\Debug\CM_1.exe (процесс 16648) завершил работу с кодом 0. Нажмите любую клавишу, чтобы закрыть это окно:`

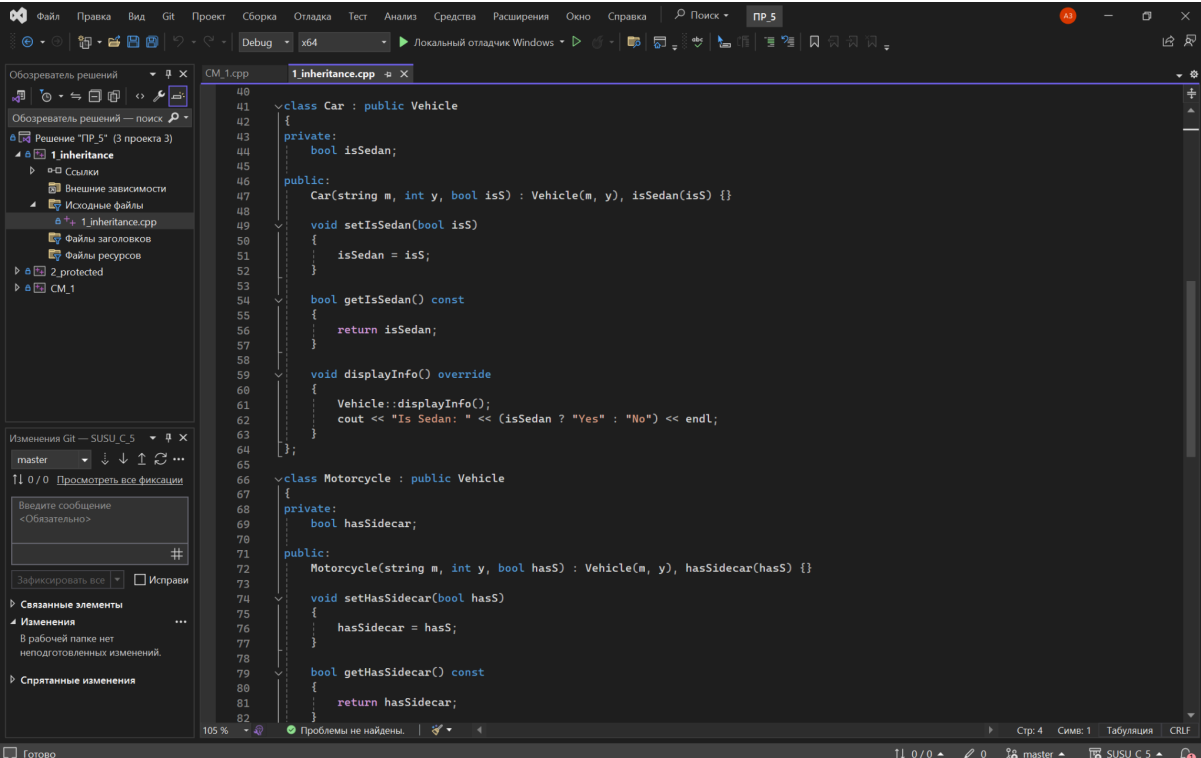
Реализация данных в файле программ

1. Наследование



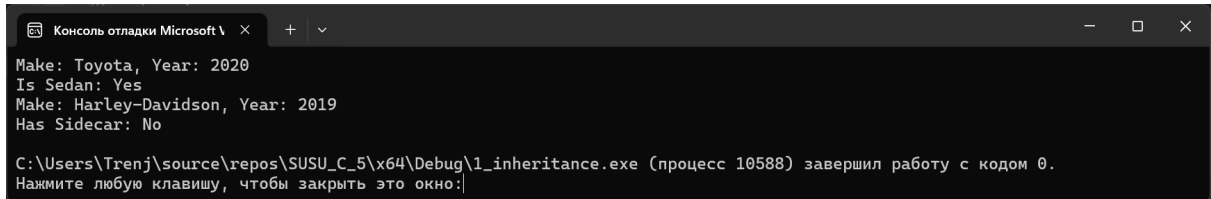
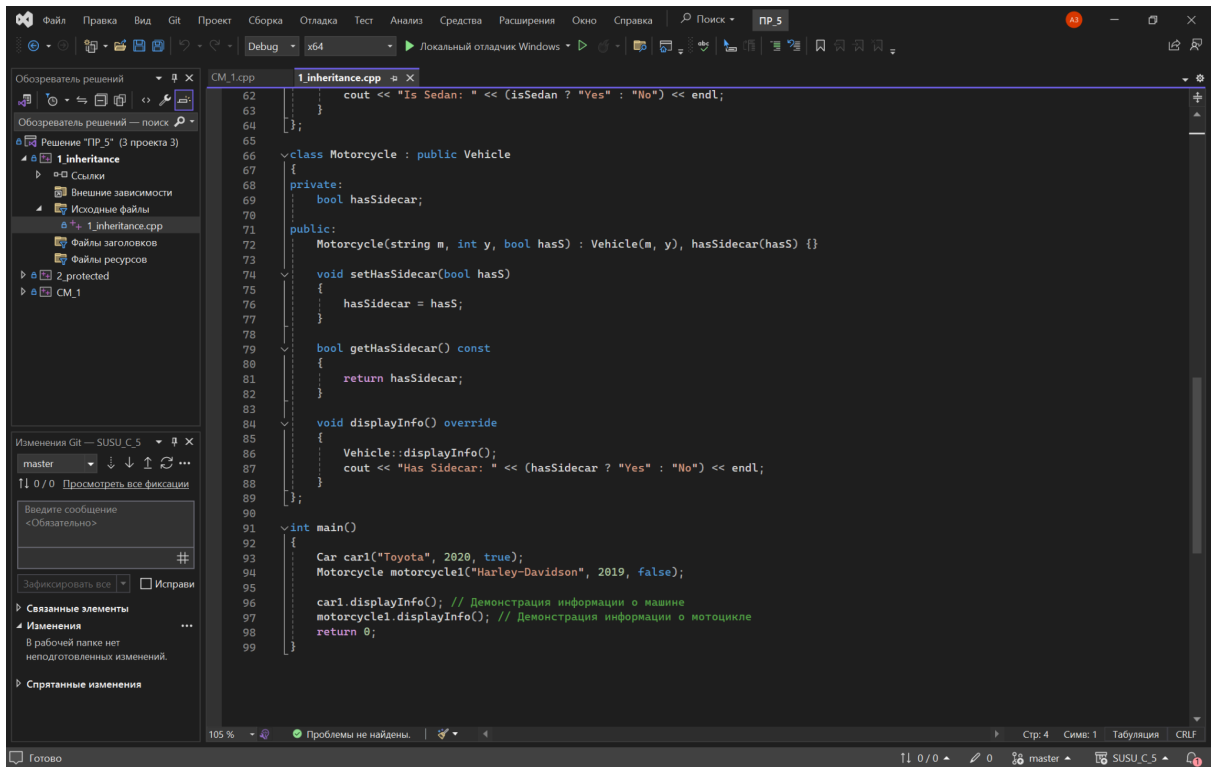
The screenshot shows the Visual Studio IDE with the file 1.inheritance.cpp open. The code defines a base class Vehicle with private attributes make and year, and public methods for setting and getting these attributes, as well as a displayInfo method. A class Car is declared as a public inheritance of Vehicle.

```
1 #include <iostream>
2 #include <string>
3 using namespace std;
4
5 class Vehicle
6 {
7 private:
8     string make;
9     int year;
10
11 public:
12     Vehicle(string m, int y) : make(m), year(y) {}
13
14     void setMake(string m)
15     {
16         make = m;
17     }
18
19     string getMake() const
20     {
21         return make;
22     }
23
24     void setYear(int y)
25     {
26         year = y;
27     }
28
29     int getYear() const
30     {
31         return year;
32     }
33
34     virtual void displayInfo()
35     {
36         cout << "Make: " << make << ", Year: " << year << endl;
37     }
38 };
39
40 class Car : public Vehicle
41 {
42 private:
```

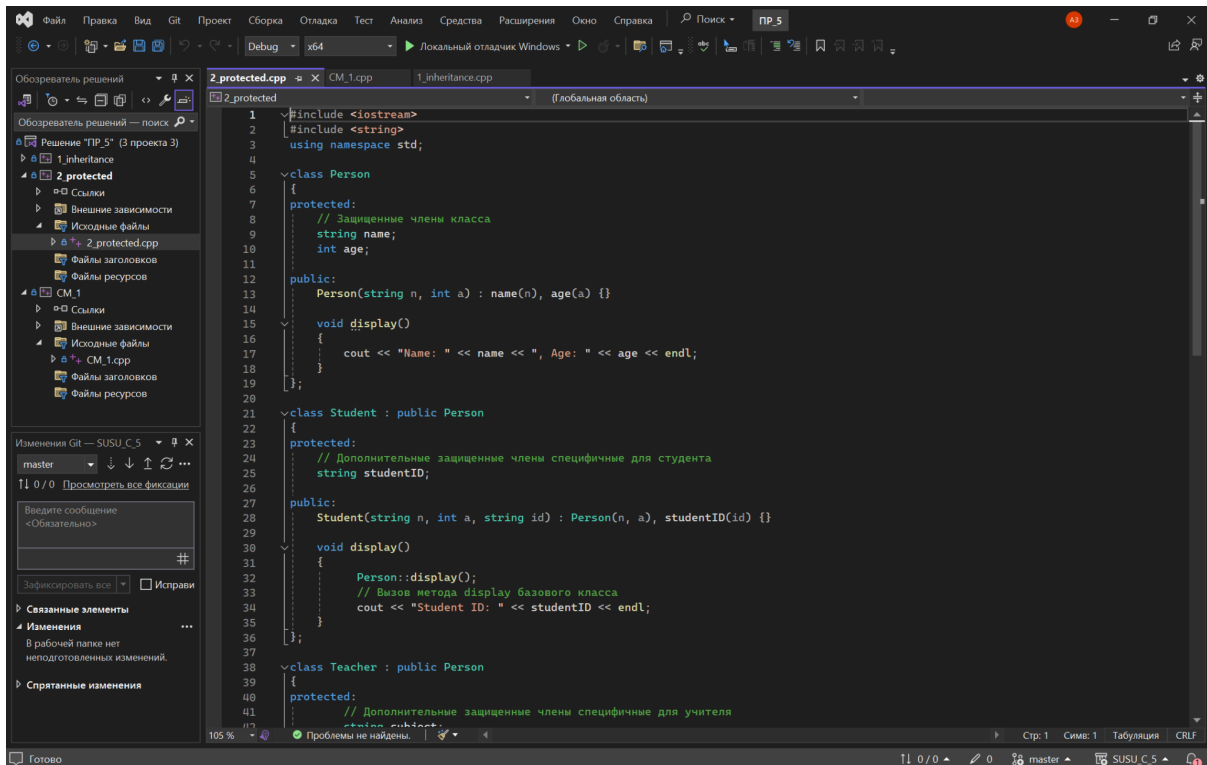


The screenshot shows the continuation of the code in 1.inheritance.cpp. The Car class is implemented with a private attribute isSedan and methods to set and get it, as well as an override of the displayInfo method. The Motorcycle class is also implemented with a private attribute hasSidecar and methods to set and get it.

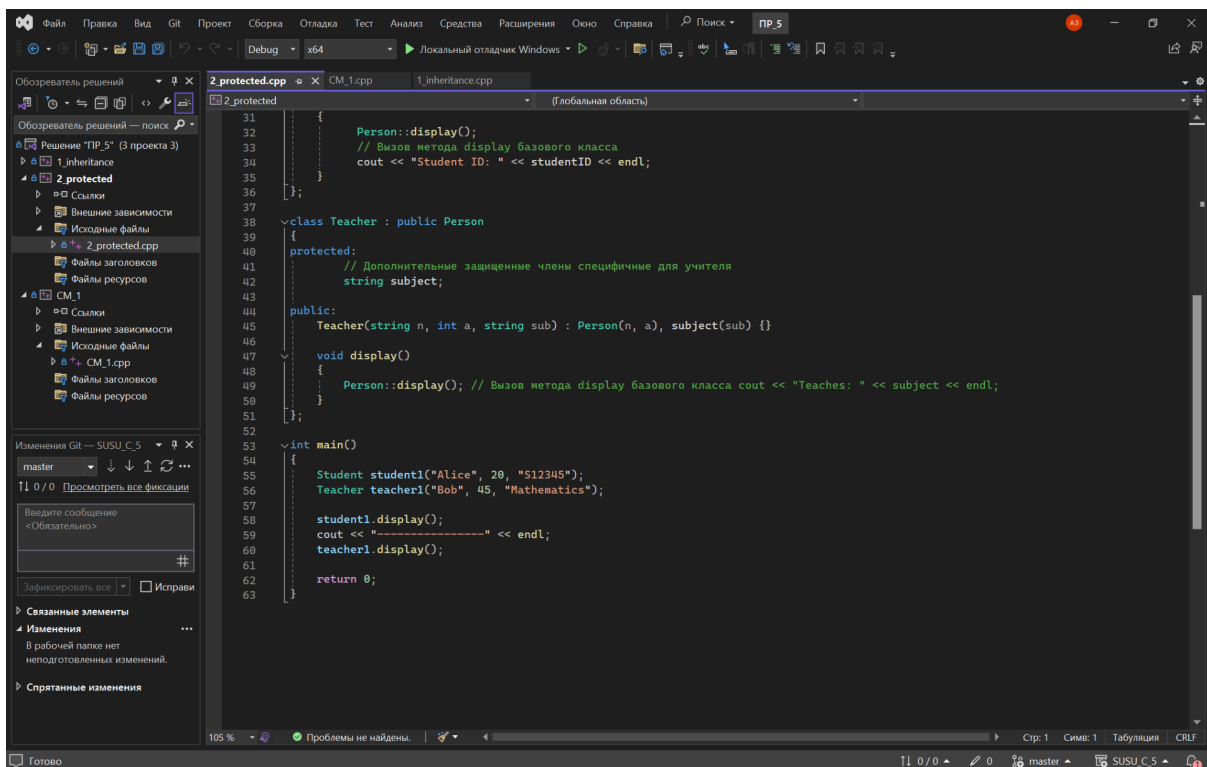
```
43 private:
44     bool isSedan;
45
46 public:
47     Car(string m, int y, bool isS) : Vehicle(m, y), isSedan(isS) {}
48
49     void setIsSedan(bool isS)
50     {
51         isSedan = isS;
52     }
53
54     bool getIsSedan() const
55     {
56         return isSedan;
57     }
58
59     void displayInfo() override
60     {
61         Vehicle::displayInfo();
62         cout << "Is Sedan: " << (isSedan ? "Yes" : "No") << endl;
63     }
64 };
65
66 class Motorcycle : public Vehicle
67 {
68 private:
69     bool hasSidecar;
70
71 public:
72     Motorcycle(string m, int y, bool hasS) : Vehicle(m, y), hasSidecar(hasS) {}
73
74     void setHasSidecar(bool hasS)
75     {
76         hasSidecar = hasS;
77     }
78
79     bool getHasSidecar() const
80     {
81         return hasSidecar;
82     }
83 }
```



2. Демонстрации использования защищенного (protected) спецификатора доступа



```
1 #include <iostream>
2 #include <string>
3 using namespace std;
4
5 class Person
6 {
7     protected:
8         // Защищенные члены класса
9         string name;
10        int age;
11
12    public:
13        Person(string n, int a) : name(n), age(a) {}
14
15        void display()
16        {
17            cout << "Name: " << name << ", Age: " << age << endl;
18        }
19
20
21    ~class Student : public Person
22    {
23        protected:
24            // Дополнительные защищенные члены специфичные для студента
25            string studentID;
26
27    public:
28        Student(string n, int a, string id) : Person(n, a), studentID(id) {}
29
30        void display()
31        {
32            Person::display();
33            // Вызов метода display базового класса
34            cout << "Student ID: " << studentID << endl;
35        }
36
37
38    ~class Teacher : public Person
39    {
40        protected:
41            // Дополнительные защищенные члены специфичные для учителя
42            string subject;
43
44    public:
45        Teacher(string n, int a, string sub) : Person(n, a), subject(sub) {}
46
47        void display()
48        {
49            Person::display(); // Вызов метода display базового класса
50            cout << "Teaches: " << subject << endl;
51        }
52
53
54    ~int main()
55    {
56        Student student1("Alice", 20, "S12345");
57        Teacher teacher1("Bob", 45, "Mathematics");
58
59        student1.display();
60        cout << "-----" << endl;
61        teacher1.display();
62
63        return 0;
64    }
```



```
31
32
33        Person::display();
34        // Вызов метода display базового класса
35        cout << "Student ID: " << studentID << endl;
36    }
37
38    ~class Teacher : public Person
39    {
40        protected:
41            // Дополнительные защищенные члены специфичные для учителя
42            string subject;
43
44    public:
45        Teacher(string n, int a, string sub) : Person(n, a), subject(sub) {}
46
47        void display()
48        {
49            Person::display(); // Вызов метода display базового класса
50            cout << "Teaches: " << subject << endl;
51        }
52
53
54    ~int main()
55    {
56        Student student1("Alice", 20, "S12345");
57        Teacher teacher1("Bob", 45, "Mathematics");
58
59        student1.display();
60        cout << "-----" << endl;
61        teacher1.display();
62
63        return 0;
64    }
```

```
Консоль отладки Microsoft \
Name: Alice, Age: 20
Student ID: S12345
-----
Name: Bob, Age: 45

C:\Users\Trenj\source\repos\SUSU_C_5\x64\Debug\2_protected.exe (процесс 3412) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно:
```