**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,**

**СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

**Санкт-Петербургский колледж телекоммуникаций им. Э.Т. Кренкеля**

**Отделение**  *Информационных технологий*

**Цикловая комиссия** *Информатики и программирования в компьютерных системах*

**Отчет о выполнении лабораторной работы**

Лабораторная работа 9. Разработка структуры классов. Создание полей и свойств, методов и конструкторов класса. Лабораторная работа 10. Разработка структуры классов. Создание операций класса.

Выполнили: Обучающийся 3 курса 500 группы,

Алексеев Василий Олегович

Проверил: преподаватель Баталов Д.И.

Санкт-Петербург

2021

1. **Цель работы**

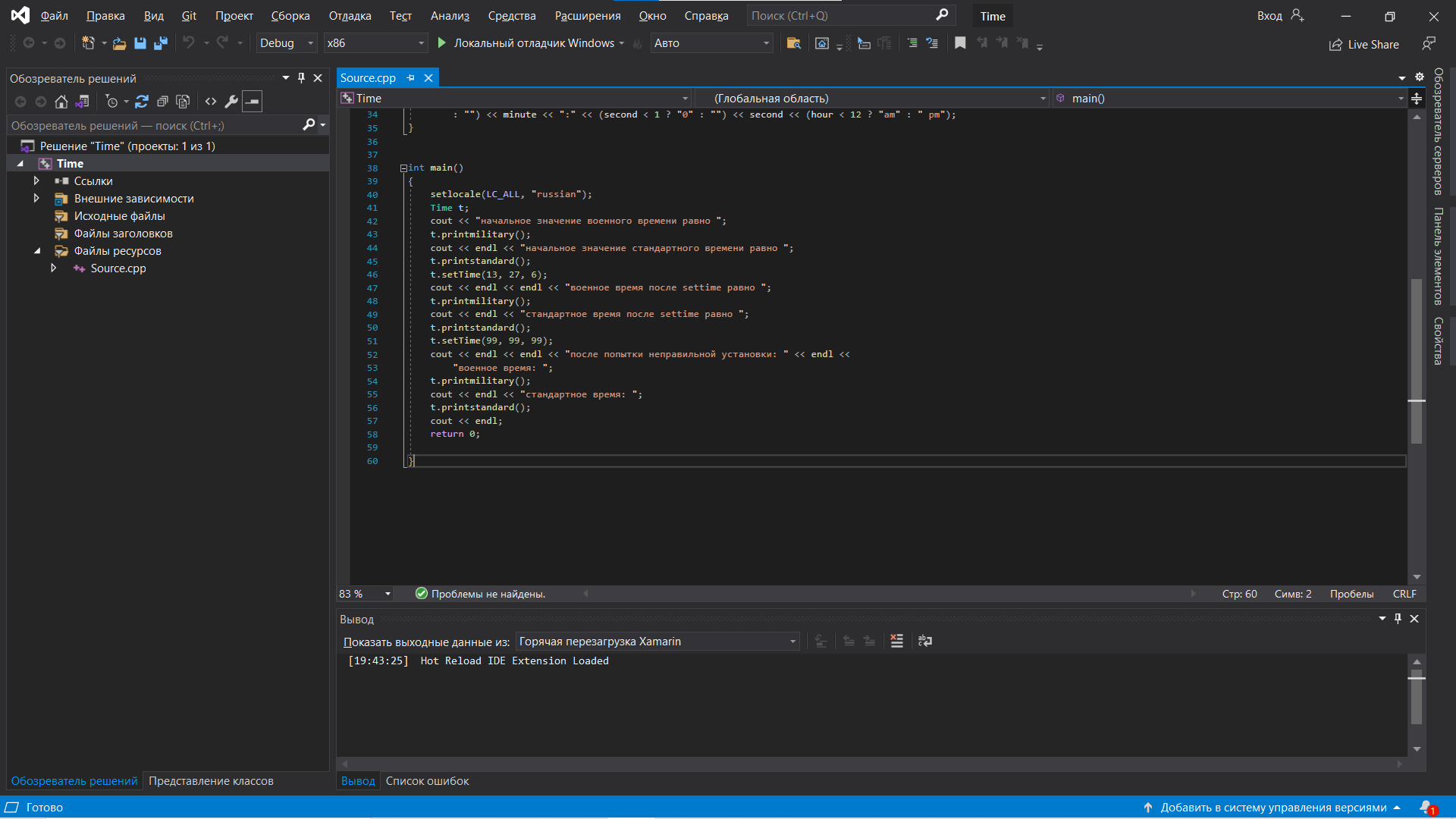
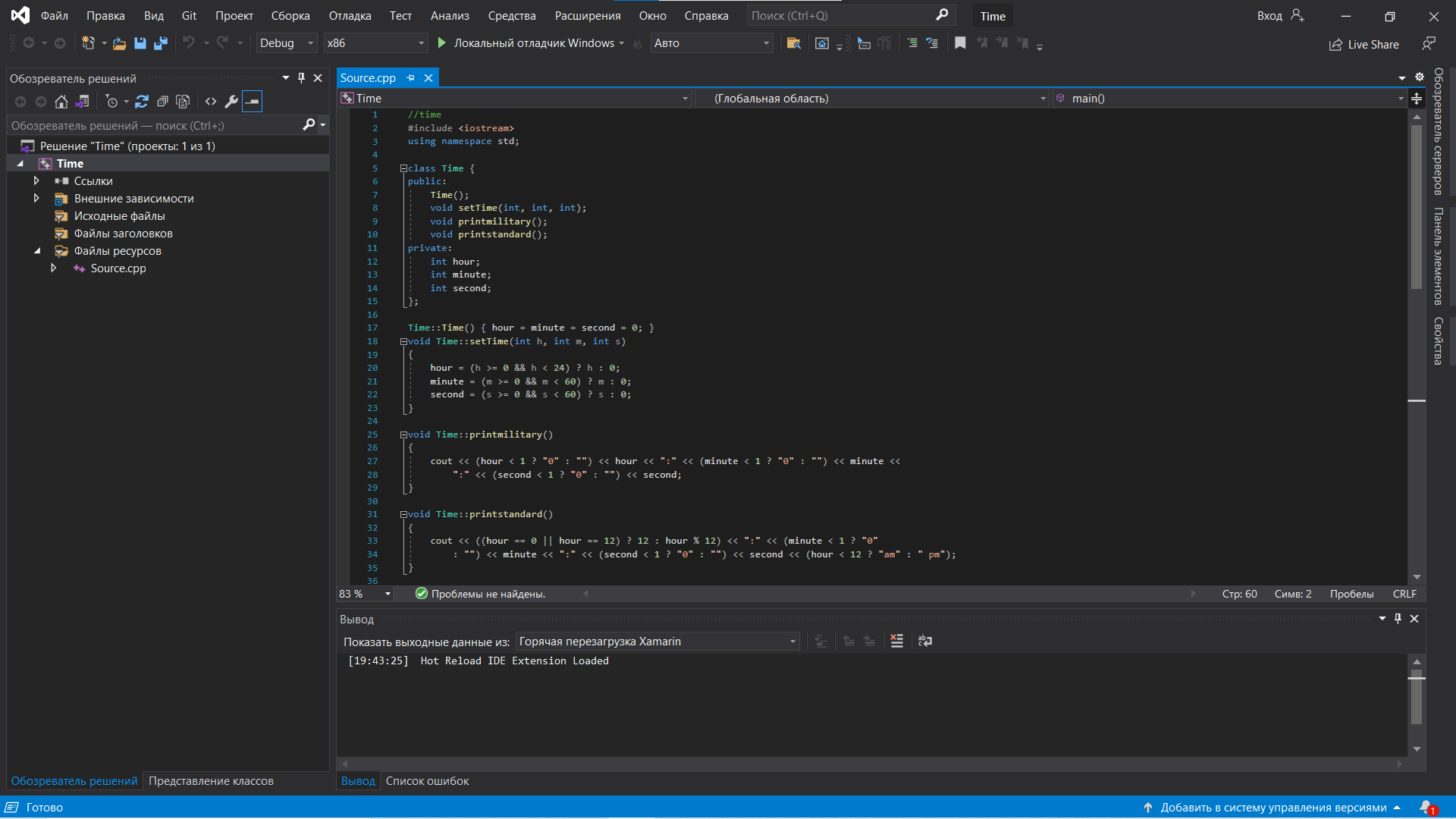
закрепление знаний о классах, составление программ с классами.

1. **Ход выполнения работы**

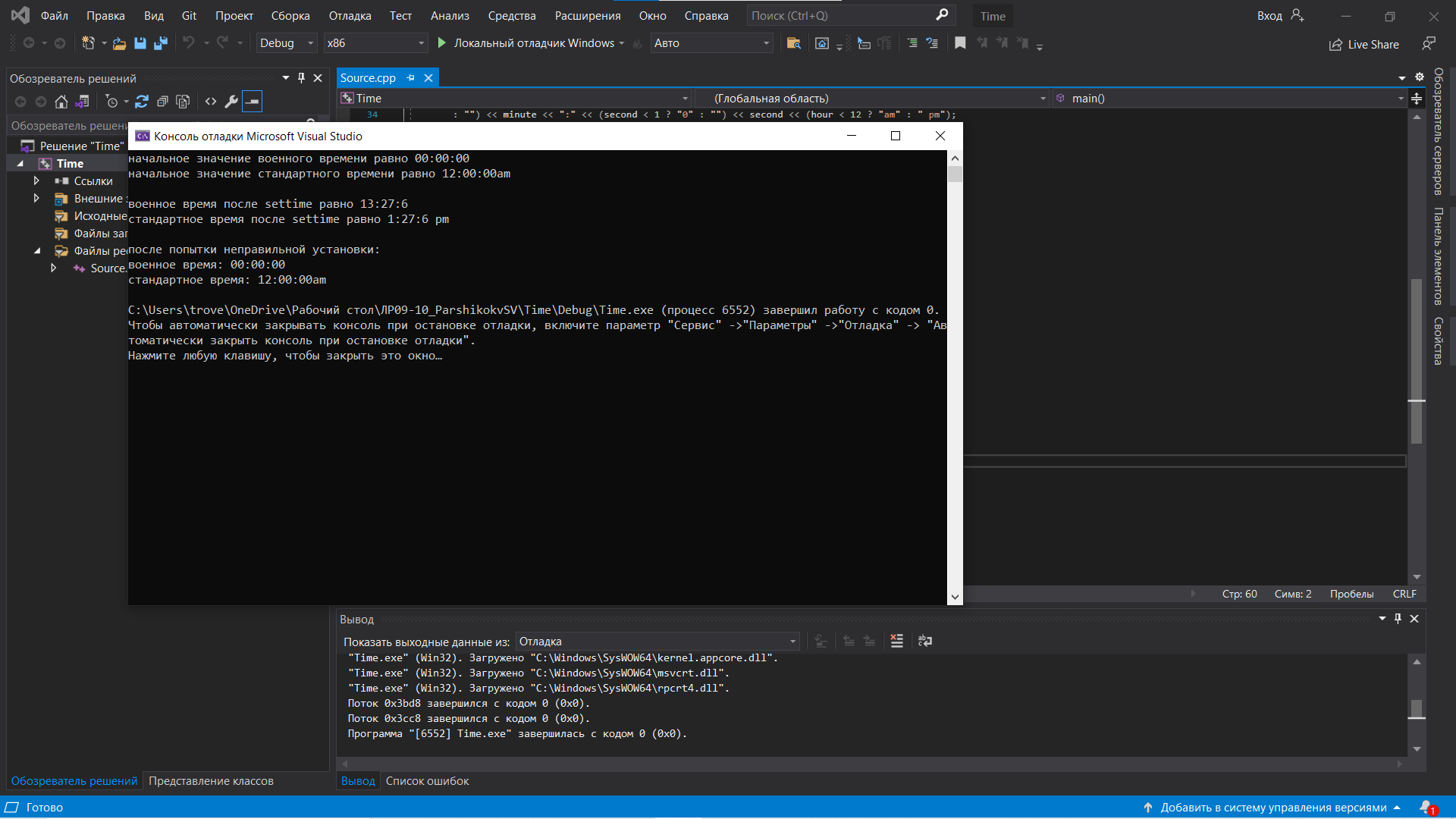
Ссылка на GitHub:

https://github.com/mazgavan/SPbCT\_AlekseevVO

1. Просмотрел приведенные примеры и проработал их.
2. Реализовал класс Time

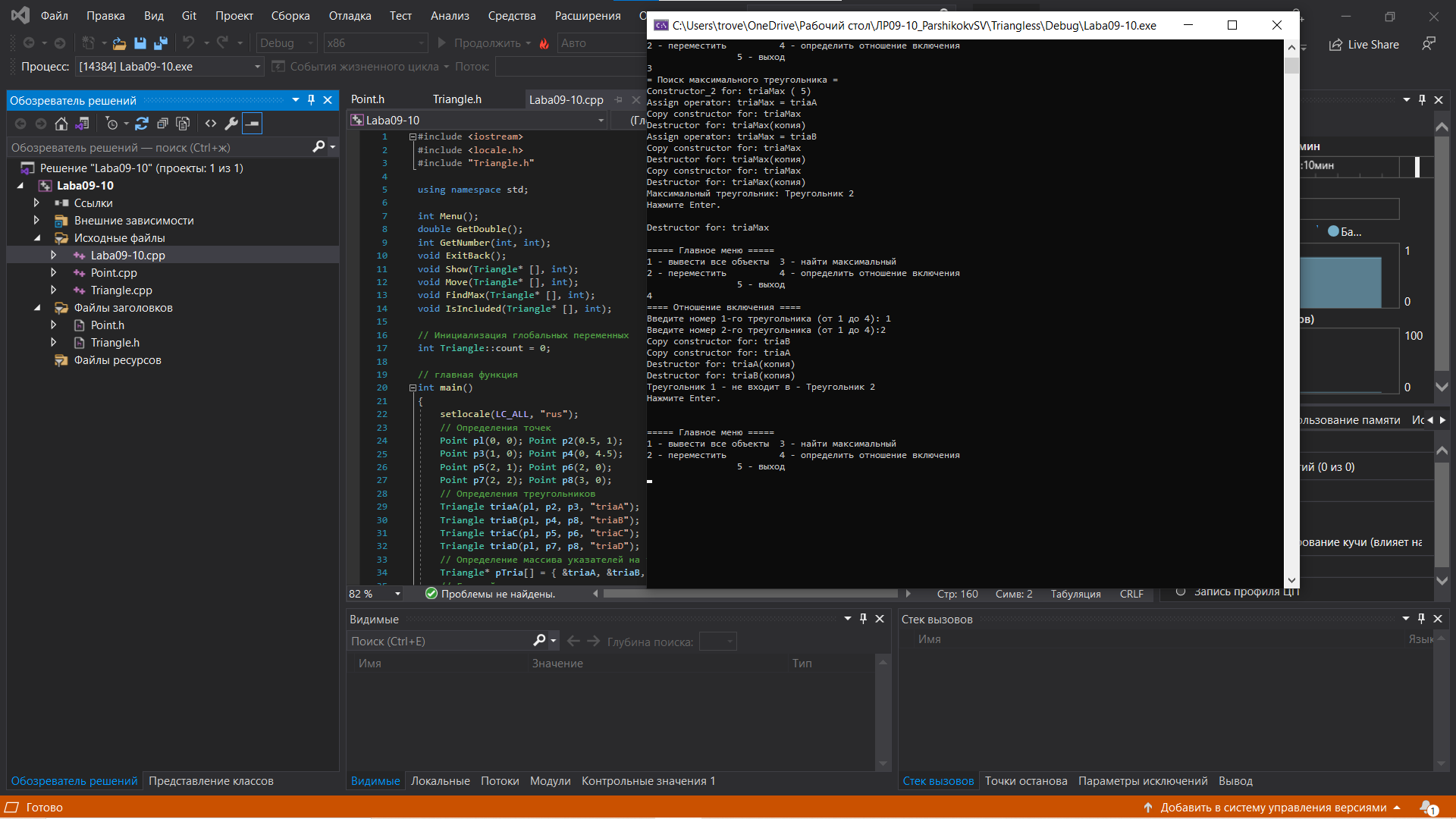
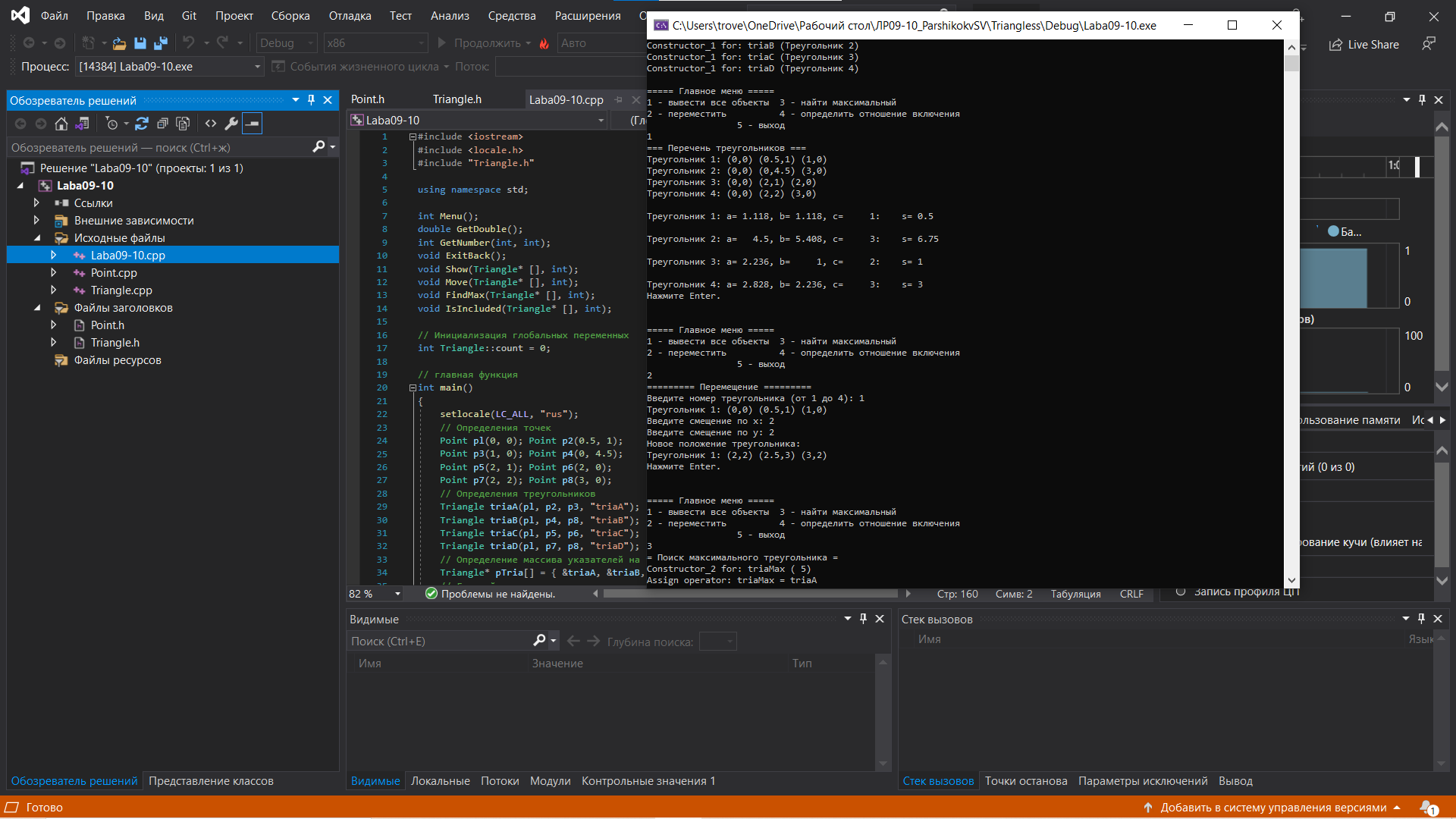


Результат:



1. Проработал класс Triangle.

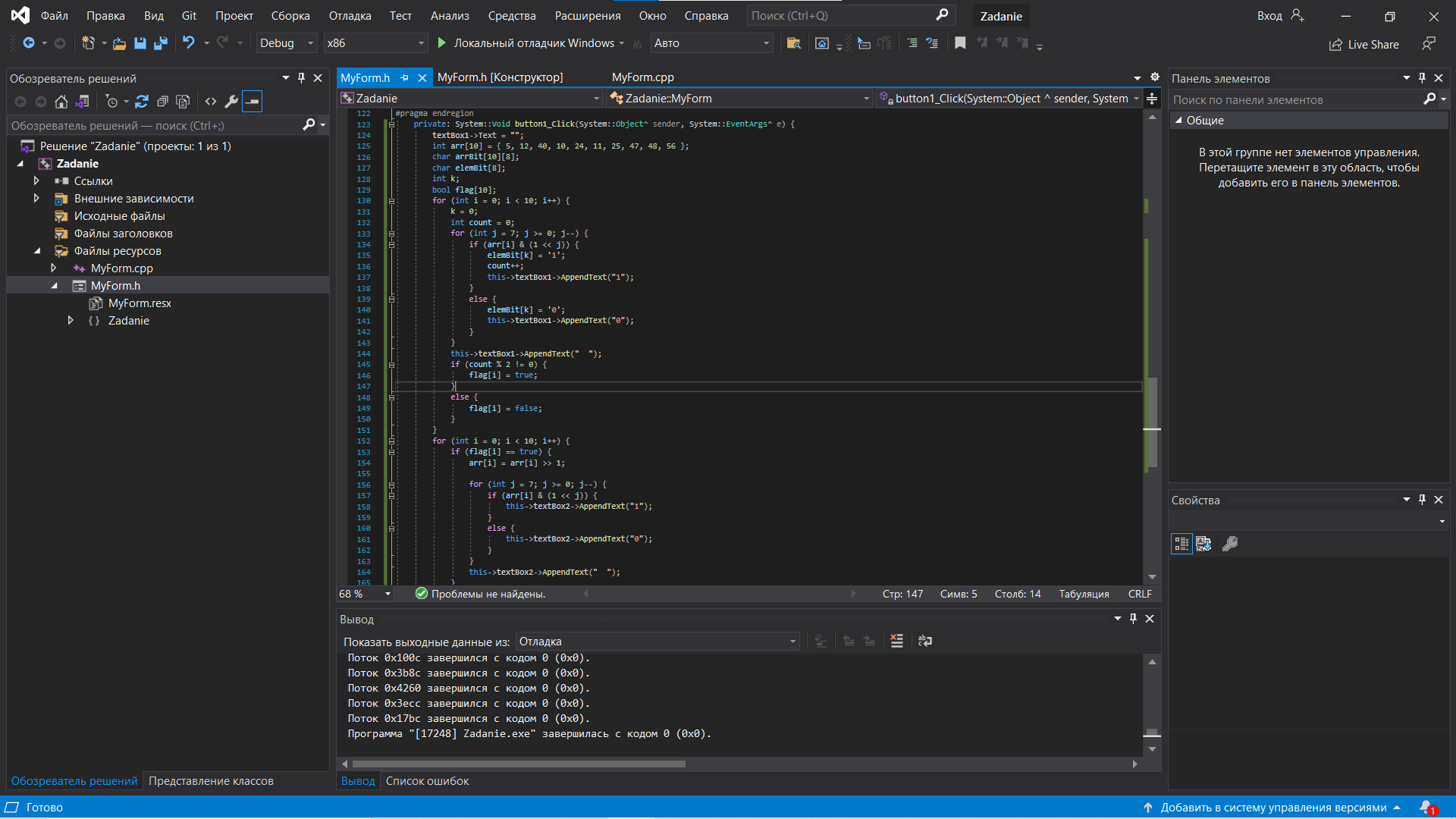
Код большой и весь есть на гитхабе, так что предоставлю только скриншот работы программы.

****

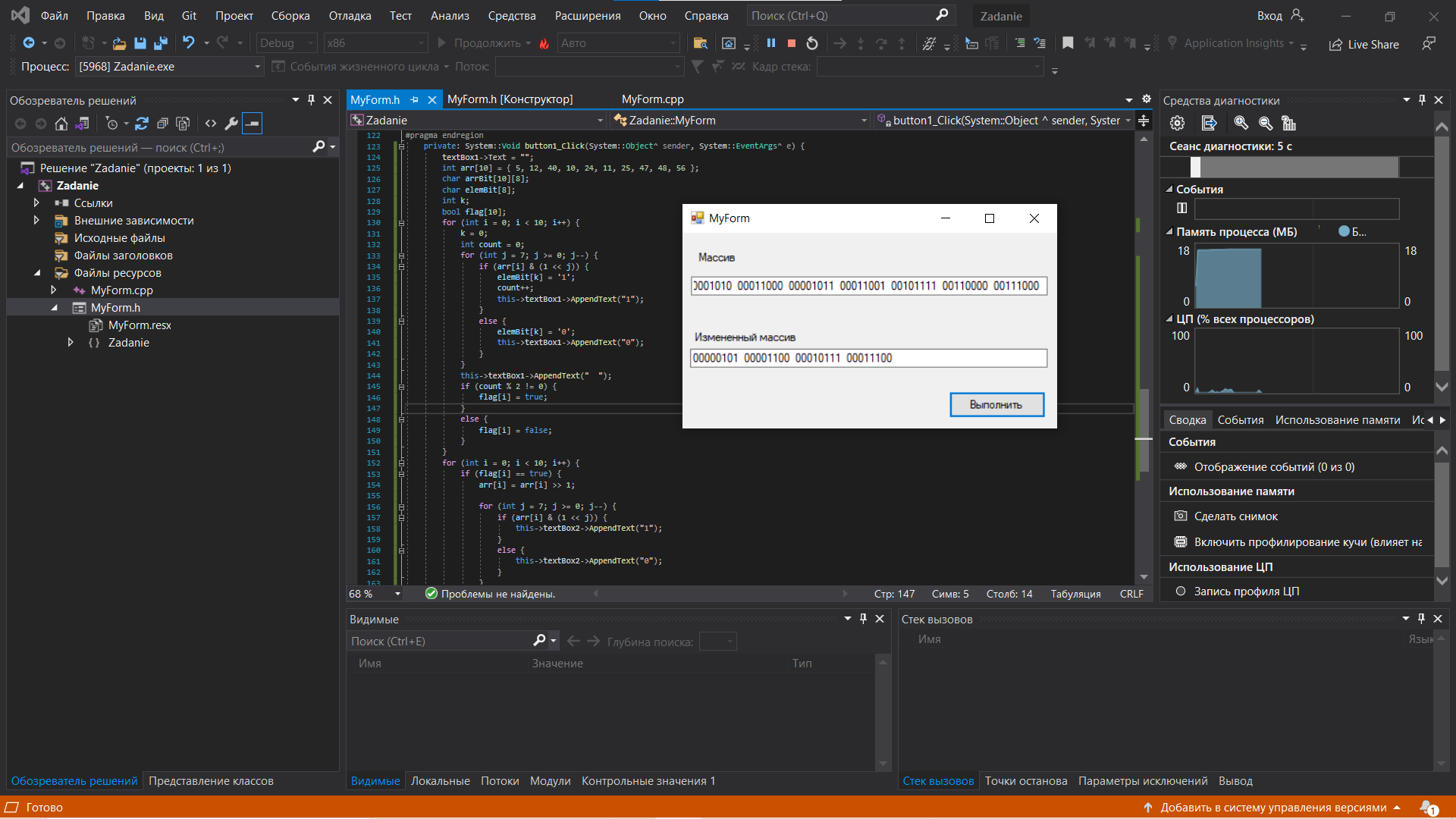
1. Выполнил задание в соответствии с вариатном.

1918007(Алексеев) + 1918157(Паршиков) = 14mod10 = 4 вариант.

Дан массив A из 10 однобайтовых чисел. Вывести на экран со сдвигом на один разряд вправо только числа, содержащие нечетное количество единиц.



Результат:



1. **Контрольные вопросы**
2. Какова структура программы на объектно-ориентированном языке?

Фундаментальные понятия объектно-ориентированного подхода могут быть воплощены по-разному при построении систем. В основе той или иной реализации системы лежит объектная модель среды, в рамках которой эта система разработана. Для программной системы первоочередное значение имеет объектная среда языка программирования. **Объектно-ориентированный язык** — это такой язык, который удовлетворяет следующим критериям:

* поддерживает объекты как абстракции данных с определенным интерфейсом поименованных операций и скрытым состоянием;
* объекты принадлежат к какому-либо типу (классу);
* типы (классы) могут наследовать атрибуты своих базовых классов.

Объектные модели разных языков программирования реализованы по-разному. Поэтому, прежде чем переходить к рассмотрению объектных моделей конкретных сред программирования, остановимся немного на критериях по которым мы будем их сравнивать. Все эти среды являются **объектно-ориентированными**. Это значит, что они обладают следующими свойствами:

1. наличием объектов и возможностью описания динамики системы как реакции на сообщения, пересылаемыми между объектами;
2. возможностью создания новых классов;
3. наличием механизма наследования для расширения или уточнения поведения имеющихся классов;
4. наличием полиморфизма между классами или объектами.

Для каждой среды программирования нужно четко представлять ее характеристики:

* **Система типов** языка определяет, какие типы предопределены в языке и каким образом можно создавать новые типы. Поскольку мы имеем дело с объектно-ориентированными языками, создание новых типов — это создание новых классов.
* **Наследование** — у каждого языка существуют свои особенности реализации механизма наследования. У всех объектно-ориентированных языков имеется возможность наследования интерфейсов и состояний.
* **Создание и уничтожение объектов** — это организация времени жизни объектов в данном языке.
* **Идентификация объектов** — это то, каким образом определяется тождество и равенство объектов.
* **Метаданные** языка описывают классы и объекты, имеющиеся в программе на данном языке. Также, рассматривается наличие **метаклассов**.
* **Модули и компоненты** языка описывают, возможность структуризации больших программ на данном языке.
* **Определение динамики поведения** объектов касается того, какова динамическая структура программы.

1. Что такое объект ООП?

Объе́ктно-ориенти́рованное программи́рование (ООП) — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования

1. Что такое класс?

В объектно-ориентированном программировании (ООП) – класс это основной элемент, в рамках которого осуществляется конструирование программ. Класс содержит в себе данные и код, который управляет этими данными.

1. На какие разделы может делиться класс?

Ключевое слово class сообщает о том, что объявляется новый класс (тип данных). Внутри класса объявляются члены класса: данные и методы. Ключевое слово private определяет члены класса, которые должны быть закрыты от внешних методов, объявленных за пределами класса, а также объектов класса. Члены данных, объявленные с ключевым словом private, доступны только другим членам этого класса.

Ключевое слово public определяет общедоступные данные (переменные) и методы (функции) класса.

Ключевое слово protected определяет защищенные данные и методы класса, которые есть:

доступными для методов унаследованных от данного класса;

недоступными для методов, реализованных в других частях программы;

недоступными для объектов (экземпляров) класса.

1. Назовите и охарактеризуйте три основных принципа в использовании классов.

Абстракция

Абстра́кция — в объектно-ориентированном программировании это придание объекту характеристик, которые отличают его от всех объектов, четко определяя его концептуальные границы. Основная идея состоит в том, чтобы отделить способ использования составных объектов данных от деталей их реализации в виде более простых объектов, подобно тому, как функциональная абстракция разделяет способ использования функции и деталей её реализации в терминах более примитивных функций, таким образом, данные обрабатываются функцией высокого уровня с помощью вызова функций низкого уровня.

Такой подход является основой объектно-ориентированного программирования. Это позволяет работать с объектами, не вдаваясь в особенности их реализации. В каждом конкретном случае применяется тот или иной подход: инкапсуляция, полиморфизм или наследование. Например, при необходимости обратиться к скрытым данным объекта, следует воспользоваться инкапсуляцией, создав, так называемую, функцию доступа или свойство.

Абстракция данных — популярная и в общем неверно определяемая техника программирования. Фундаментальная идея состоит в разделении несущественных деталей реализации подпрограммы и характеристик существенных для корректного ее использования. Такое разделение может быть выражено через специальный «интерфейс», сосредотачивающий описание всех возможных применений программы [1].

С точки зрения теории множеств, процесс представляет собой организацию для группы подмножеств своего множества. См. также Закон обратного отношения между содержанием и объемом понятия.

Инкапсуляция

Инкапсуляция — свойство программирования, позволяющее пользователю не задумываться о сложности реализации используемого программного компонента (что у него внутри?), а взаимодействовать с ним посредством предоставляемого интерфейса (публичных методов и членов), а также объединить и защитить жизненно важные для компонента данные. При этом пользователю предоставляется только спецификация (интерфейс) объекта.

Пользователь может взаимодействовать с объектом только через этот интерфейс. Реализуется с помощью ключевого слова: public.

Пользователь не может использовать закрытые данные и методы. Реализуется с помощью ключевых слов: private, protected, internal.))

Инкапсуляция — один из четырёх важнейших механизмов объектно-ориентированного программирования (наряду с абстракцией, полиморфизмом и наследованием).

Сокрытие реализации целесообразно применять в следующих случаях:

предельная локализация изменений при необходимости таких изменений,

прогнозируемость изменений (какие изменения в коде надо сделать для заданного изменения функциональности) и прогнозируемость последствий изменений.

Наследование

Наследование — один из четырёх важнейших механизмов объектно-ориентированного программирования (наряду с инкапсуляцией, полиморфизмом и абстракцией), позволяющий описать новый класс на основе уже существующего (родительского), при этом свойства и функциональность родительского класса заимствуются новым классом.

Другими словами, класс-наследник реализует спецификацию уже существующего класса (базовый класс). Это позволяет обращаться с объектами класса-наследника точно так же, как с объектами базового класса.

1. Что такое конструктор и для чего он используется?

Конструктор - это специальный метод, который вызывается при создании нового объекта. Не всегда удобно инициализировать все переменные класса при создании его экземпляра. Иногда проще, чтобы какие-то значения были бы созданы по умолчанию при создании объекта.

1. Как происходит индексирование объектов класса?

Чтобы задать смысл индексов для объектов класса используется

функция operator[]. Второй параметр (индекс) функции operator[] может быть любого типа.

Это позволяет определять ассоциативные массивы и т.п.

1. Как происходит вызов функции?

Вызов функции, то есть запись выражение(список\_выражений),

можно проинтерпретировать как бинарную операцию, и операцию вызова можно перегружать так же, как и другие операции. Список параметров функции operator() вычисляется и проверяется в соответствие с обычными правилами передачи параметров. Перегружающая функция может оказаться полезной главным образом для определения типов с единственной операцией и для типов, у которых одна операция настолько преобладает, что другие в большинстве ситуаций можно не принимать во внимание.

1. Что такое композиция классов?

Включение нескольких объектов других классов в данный класс

с тем, чтобы данный класс мог брать нужные сведения из других классов называется

композицией.

1. Для чего и каким образом используются дружественные функции?

Дружественные функции определяются вне области действия этого класса, но имеют право доступа к закрытым элементам private данного класса. Функция или класс в целом могут быть объявлены другом (friend) другого класса.

Дружественные функции используются для повышения производительности.

1. Для чего требуется указатель this?

Когда функция элемент ссылается на другой элемент какого-то объекта данного класса, имеется ввиду соответствующий объект. Это происходит благодаря тому, что каждый объект сопровождается указателем на самого себя - называемым указателем this - это неявный аргумент во всех ссылках на элементы внутри этого объекта. Указатель this можно использовать также и явно. Каждый объект может определить свой собственный адрес с помощью ключевого слова this.

1. Для чего используется модификатор static?

Модификатор static позволяет объявить поле в классе, которое будет общим для всехэкземпляров класса. Кроме объявления статического поля в классе, необходимо также дать егоопределение в глобальной области видимости программы.