**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 11. Шаблоны функций. Лабораторная работа 12. Шаблоны классов.

Выполнили: Обучающиеся 3 курса 500 группы,

Паршиков Сергей Витальевич,

Алексеев Василий Олегович.

Проверил: преподаватель Баталов Д.И.

Санкт-Петербург

2021

1. **Цель работы**

Целью лабораторной работы является получение

практических навыков создания абстрактных типов данных и перегрузки операций в языке

С++, создания шаблонов абстрактных типов данных и использования их в программах на С++

1. **Ход выполнения работы**

1918007(Алексеев) + 1918157(Паршиков) = 14mod12 = 2 вариант

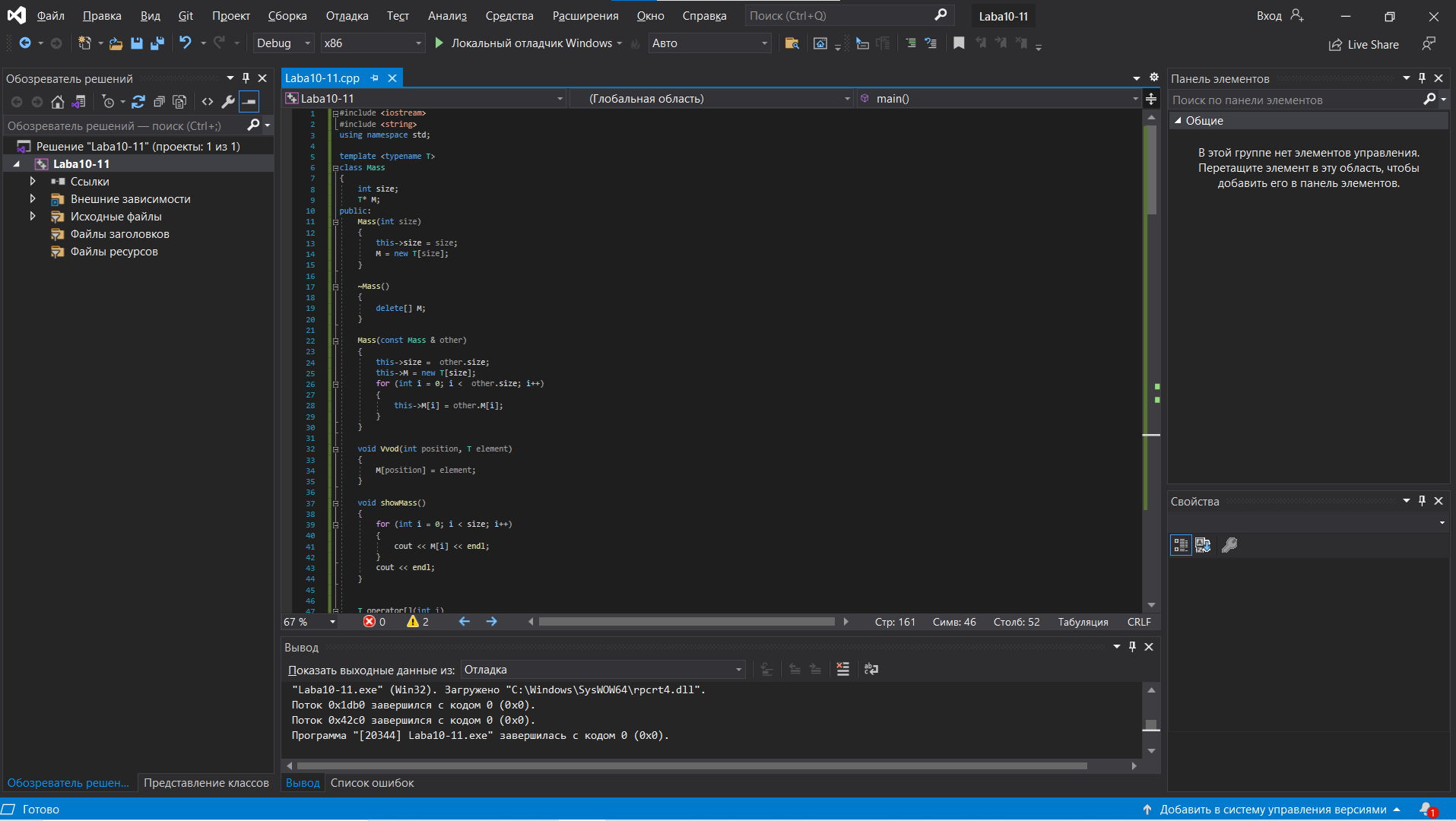
Класс - одномерный массив. Дополнительно перегрузить следующие операции:

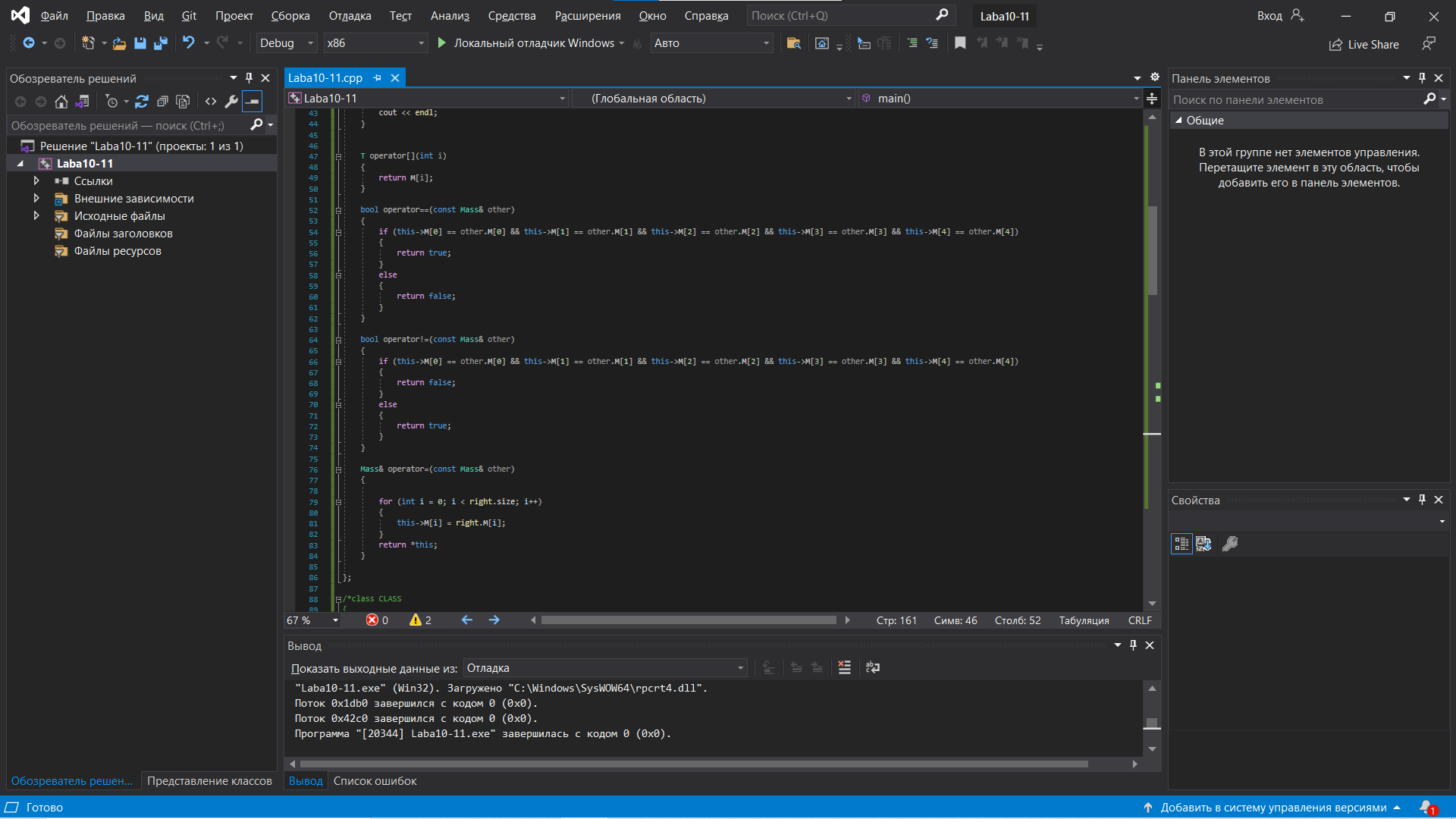
[] - доступ по индексу;

= = - проверка на равенство;

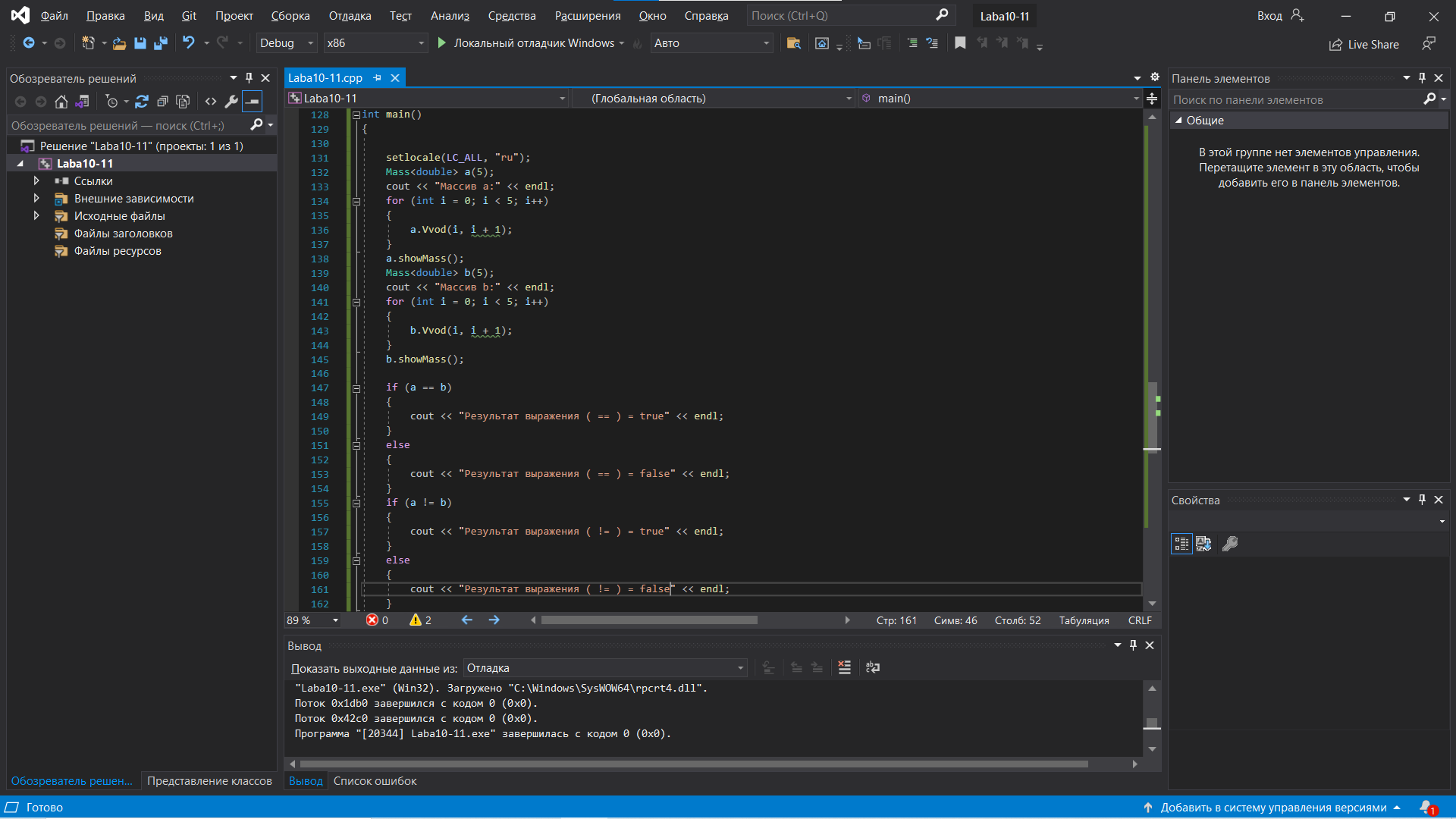
!= - проверка на неравенство.

1. Создали шаблон заданного класса, в соответствии с вариантом. Определили конструкторы, деструктор, перегруженную операцию присваивания (“=”) и операции, заданные в варианте задания.

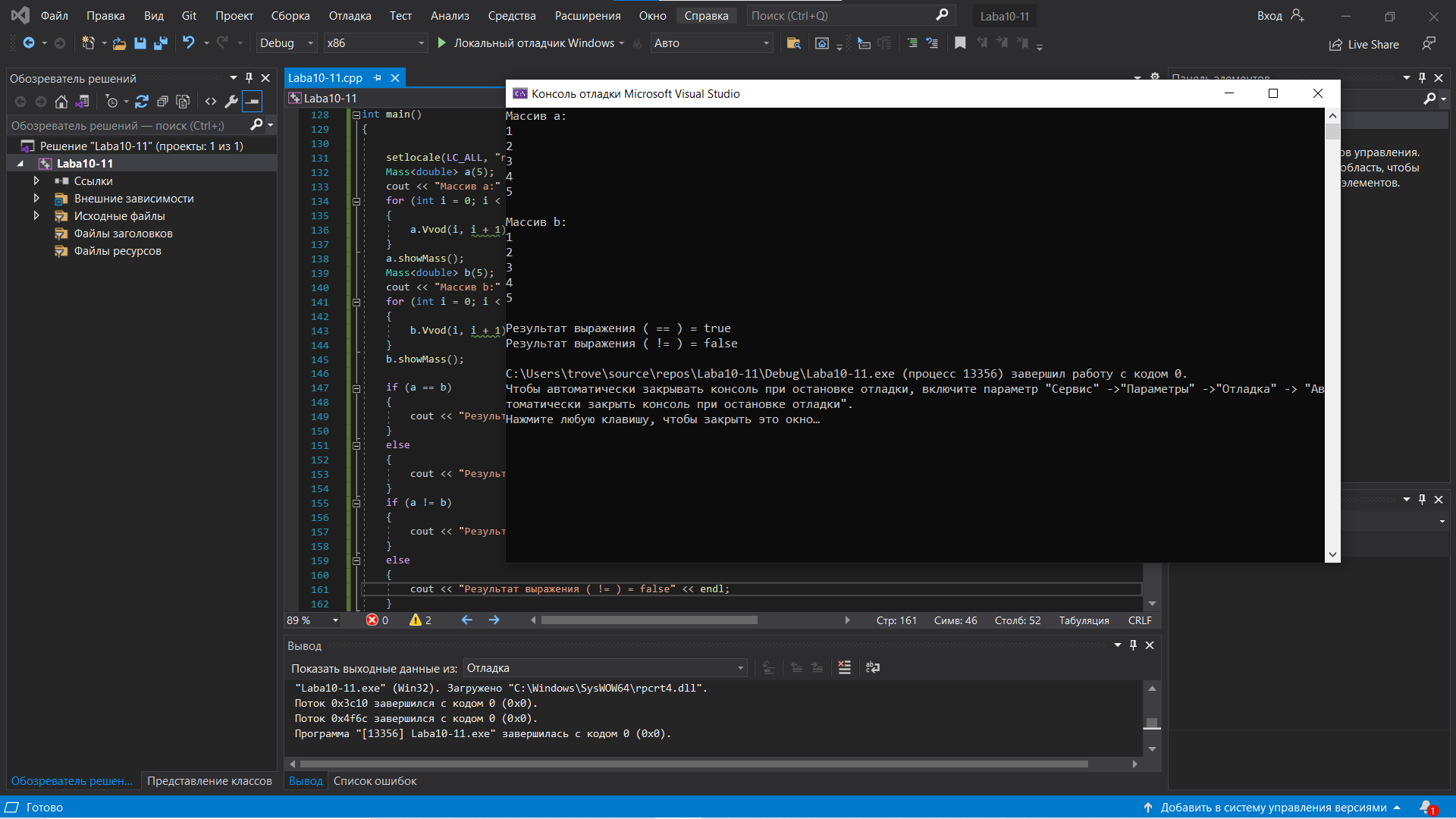




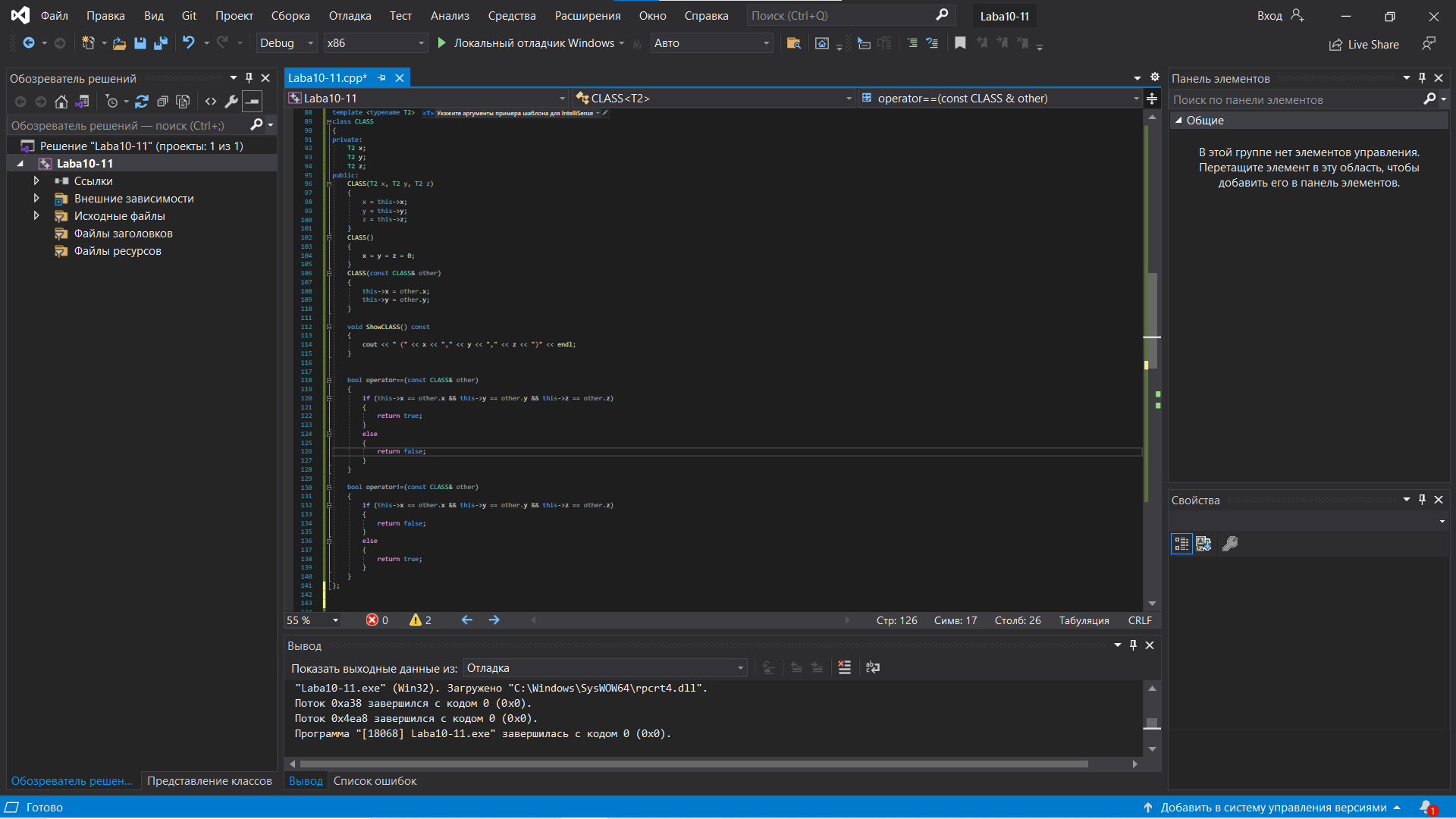
1. Написали программу тестирования, в которой проверяется использование шаблона для стандартных типов данных.



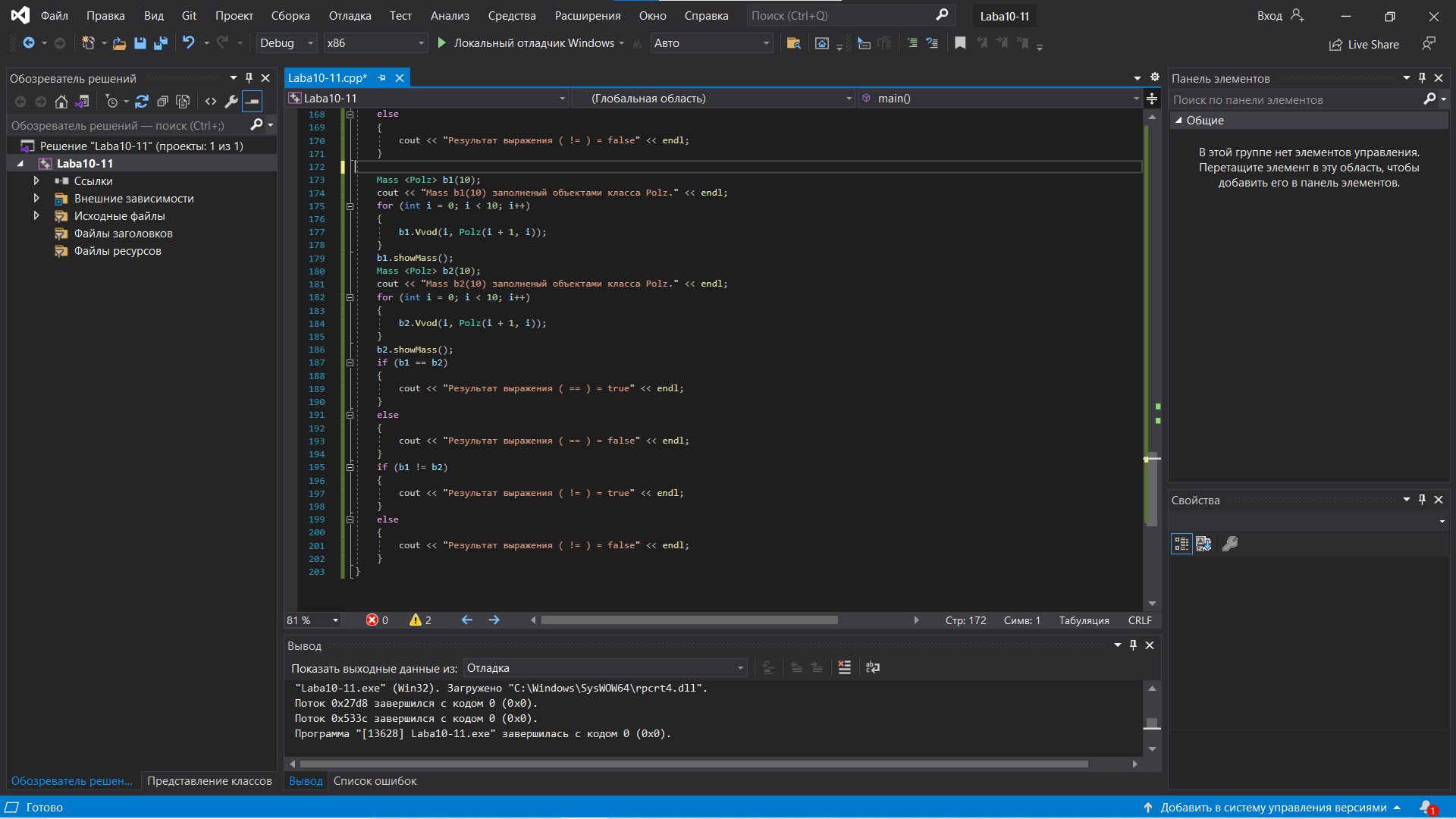
1. Выполнили тестирование.



4. Определили пользовательский класс, который будет использоваться в качестве параметра шаблона. Определили в классе необходимые функции и перегруженные операции.



5. Написали программу тестирования, в которой проверяется использование шаблона для пользовательского типа.



6. Выполнили тестирование.

Неизвестно почему выдает ошибку(.

1. **Контрольные вопросы**
2. В чем смысл использования шаблонов?

В языке C++ шаблоны функций — это функции, которые служат образцом для создания других подобных функций. Главная идея — создание функций без указания точного типа(ов) некоторых или всех переменных. Для этого мы определяем функцию, указывая тип параметра шаблона, который используется вместо любого типа данных.

1. Синтаксис/семантика шаблонов функций?

Шаблон — это конструкция, которая создает обычный тип или функцию во время компиляции на основе аргументов, предоставленных пользователем для параметров шаблона. Например, можно определить шаблон функции следующим образом:

template <typename T>

T minimum(const T& lhs, const T& rhs)

{

return lhs < rhs ? lhs : rhs;

}

1. Синтаксис/семантика шаблонов классов?

Если нам надо создать шаблон класса, с одним параметром типа int и char, шаблон класса будет выглядеть так:

template <typename T>

class Name

{

//тело шаблона класса

};

где T — это параметр шаблона класса, который может принимать любой из встроенных типов данных, то, что нам и нужно.

1. Определить параметризованную функцию сортировки массива методом обмена.

Параметризованная функция обмена двух переменных

template void Swap(Stype &f,Stype &g)

{

Stype temp; //ВместоStypeбудет подставлен реальный тип данных

temp=f;

f=g;

g=temp;

}

1. Определить шаблон класса “вектор” - одномерный массив.

Ответ не нашли.

1. Что такое параметры шаблона функции?

Шаблоны позволяют создавать параметризованные классы и функции. Параметром может быть любой тип или значение одного из допустимых типов (целое число, enum, указатель на любой объект с глобально доступным именем, ссылка).

1. Перечислите основные свойства параметров шаблона функции.

1. Имена параметров шаблона должны быть уникальными во всем определении шаблона.

2. Список параметров шаблона функции не может быть пустым, так как при этом теряется возможность параметризации и шаблон функций становится обычным определением конкретной функции.

3. В списке параметров шаблона функций может быть несколько параметров. Каждый из них должен начинаться со служебного слова class. Например, допустим такой заголовок шаблона:

template <class type1, class type2>

Соответственно, неверен заголовок:

template <class type1, type2, type3>

4. Недопустимо использовать в заголовке шаблона параметры с одинаковыми именами.

1. Можно ли перегружать параметризованные функции?

Да, меняя тип данных параметра, либо меняя параметры местами, в том случае, если они разного типа, а также можно выполнять перегрузку функции добавляя или исключая параметры.

1. Перечислите основные свойства параметризованных классов.

Компонентные функции параметризованного класса автоматически являются параметризованными. Их не обязательно объявлять как параметризованные с помощью template.

Дружественные функции, которые описываются в параметризованном классе, не являются автоматически параметризованными функциями, т.е. по умолчанию такие функции являются дружественными для всех классов, которые организуются по данному шаблону.

1. Может ли быть пустым список параметров шаблона? Объясните.

Список параметров шаблона не может быть пустым. В списке параметров шаблона может быть несколько параметров, и каждому из них должно предшествовать ключевое слово class. ... Все параметры шаблона функций должны быть обязательно использованы в спецификациях параметров определения функции.

1. Как вызвать параметризованную функцию без параметров?

class SomeClass{

int SomeValue;

int SomeArray[20];

...

};

1. Все ли компонентные функции параметризованного класса являются

параметризованными?

Компонентные функции параметризованного класса автоматически являются параметризованными. Их не обязательно объявлять как параметризованные с помощью template. Дружественные функции, которые описываются в параметризованном классе, не являются автоматически параметризованными функциями

1. Являются ли дружественные функции, описанные в параметризованном классе, параметризованными?

Дружественные функции, которые описываются в параметризованном классе, не являются автоматически параметризованными функциями, т. е. по умолчанию такие функции являются дружественными для всех классов, которые организуются по данному шаблону.

1. Могут ли шаблоны классов содержать виртуальные компонентные функции?

Локальные классы не могут содержать шаблоны в качестве своих

элементов

1. Как определяются компонентные функции параметризованных классов вне определения шаблона класса?

Не нашли ответ

1. Каковы синтаксис/семантика “операции-функции”?

Формат определения операции-функции:

Тип возвращаемого значения operator знак\_ операции (

спецификация формальных параметров)

{тело операции-функции

}

1. Когда нужно перегружать операцию присваивания для определенного

пользователем типа данных, например класса?

Перегружать можно только операции, для которых хотя бы один аргумент представляет тип данных, определенный пользователем. Функция-операция должна быть определена либо как функция-член класса, либо как внешняя функция, но дружественная классу.

1. Можно ли изменить приоритет перегруженной операции?

Имеются некоторые ограничения на перегрузку операторов. Во-первых, нельзя изменить приоритет оператора. Во-вторых, нельзя изменить число операндов оператора. Наконец, за исключением оператора присваивания, перегруженные операторы наследуются любым производным классом.

1. Можно ли изменить количество операндов перегруженной операции?

Нельзя. Ответ в предыдущем вопросе.

20. Можно ли, используя дружественную функцию, перегрузить оператор

присваивания?

Не всё может быть перегружено через дружественные функции

Операторы присваивания ( = ), индекса ( [] ), вызова функции ( () ) и выбора члена ( -> ) перегружаются через методы класса — это требование языка C++.

1. Все ли операции языка С++ могут быть перегружены?

Не все операции языка С++ могут быть перегружены. Нельзя перегрузить следующие операции: ... Каждая операция, заданная в языке, имеет определенное число операндов, свой приоритет и ассоциативность.

22. Все ли операции можно перегрузить с помощью глобальной дружественной

функции?

Не нашли ответ

1. В каких случаях операцию можно перегрузить только глобальной функцией?

когда левый операнд должен быть не экземпляр нашего класса, а что-то иного типа, тогда и делают глобальной, впрочем, ничего не мешает сделать оператор глобальной ф. даже с левым операндом

1. В каких случаях глобальная операция-функция должна быть дружественной?

очевидно, если для её работы необходим доступ к приватным полям и методам класса.

С самой темой перегрузки операторов дружественность никак не связана. Просто так совпадает, что глобальные операторы делают друзьями, чтоб не париться с доступом к данным из них, но они вполне могут и не быть дружественными, а для доступа (скажем) использовать геттеры.