Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Пермский национальный исследовательский политехнический университет» (ПНИПУ)

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

**Лабораторная работа по классам № 3**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

**семестр** 2

Выполнил студент гр. ИВТ-21-1б

Балдин Алексей Игоревич

Проверил:

Яруллин Денис Владимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

г. Пермь - 2022

**Цель**

Использование перегруженных операций в классах.

**Постановка задачи**

Определить пользовательский класс.

Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.

Определить в классе деструктор.

Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы).

Перегрузить операцию присваивания.

Перегрузить операции ввода и вывода объектов с помощью потоков.

Перегрузить операции, указанные в варианте.

Написать программу, в которой продемонстрировать создание объектов и работу всех перегруженных операций.

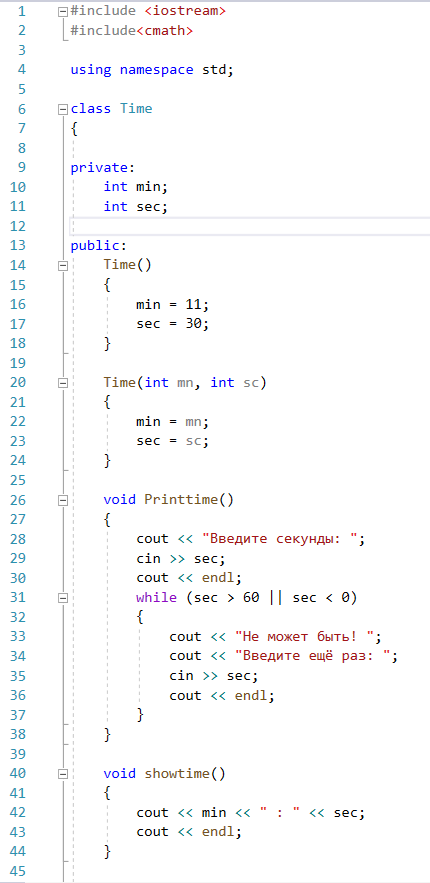
**Исходные данные**

Вариант 5:

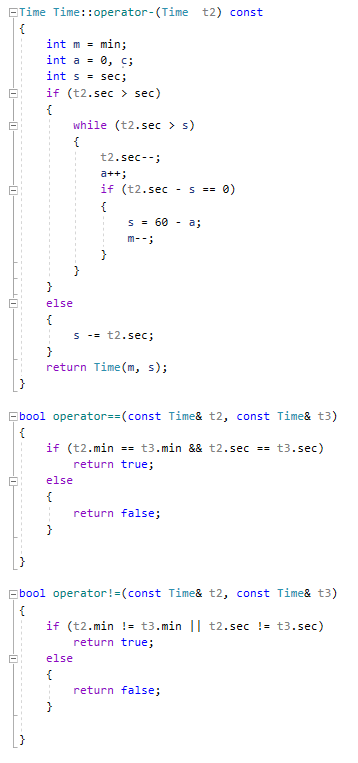
Создать класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты отделяются от секунд двоеточием. Реализовать:

* вычитание секунд
* сравнение временных интервалов (== и !=)

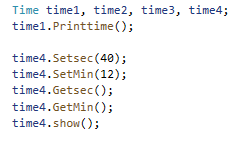
Сначала создаём тип данных «class» под именем Time. Поля классов min и sec c типом данных (int), находятся в private. В модификаторе доступа – public, создаем конструкторы по умолчанию, без параметров, деструктор, также метод – Printtime, который как раз-таки вводит секунды, также метод showtime – выводит минуты и секунды, еще методы (геттеры и сеттеры) – т.е. установка значений в private поля, еще метод showsec – который показывает секунды. Тут же в public прописываем дружественность операторов (==) и (!=).



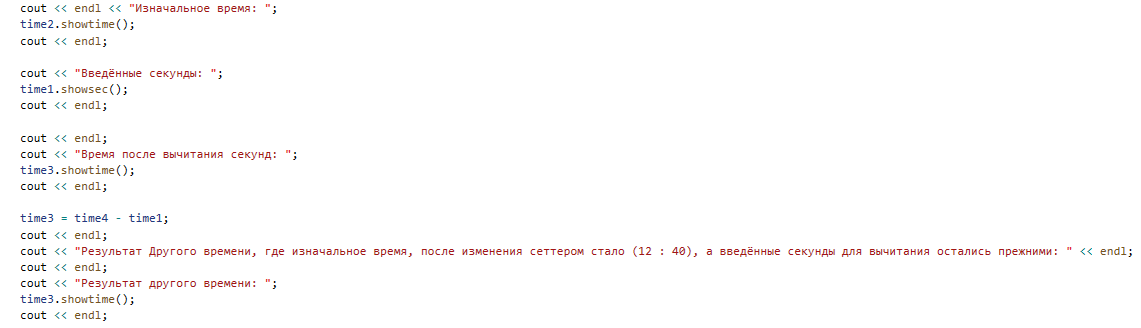
Здесь, а именно «глобально» мы описываем работу наших операторов в соответствии с условем задачи:



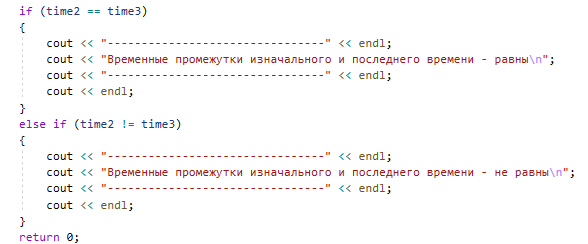
В int main () создаю объекты типа Time( т е. нашего класса), первый объект будет принимать значения по умолчанию, объект 4 – будет принимать значения с помощью методов класса(12 мин 40 сек).



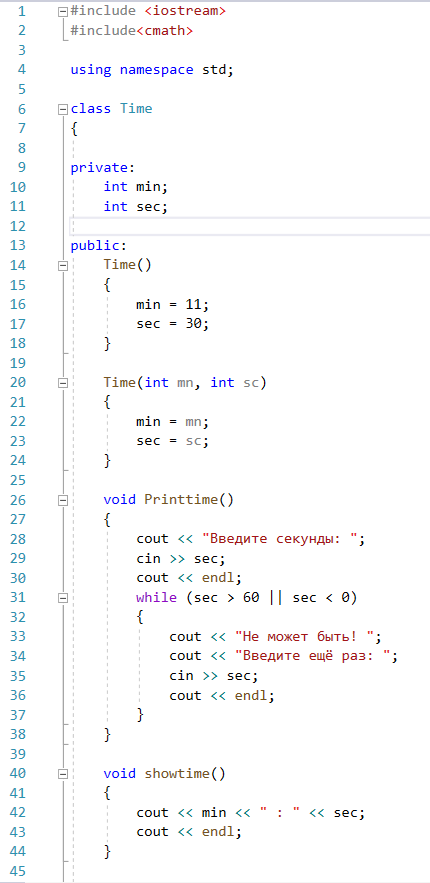
Объект 2 - вывожу его поля, объект 1 – вывожу секунды, объект 3 – хранит в себе разность объекта 4 и объекта 1. После вывожу результат.

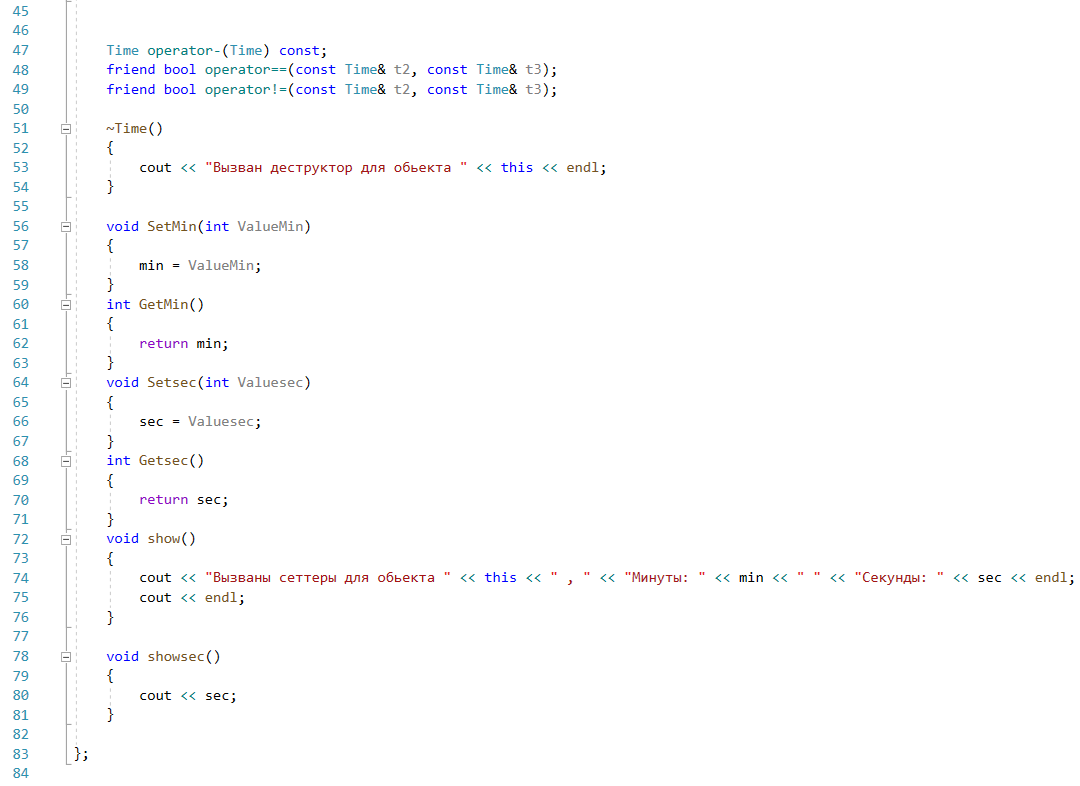


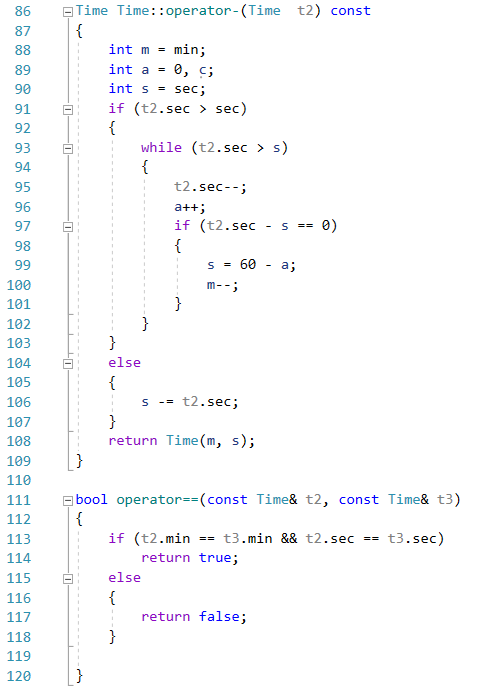
Здесь просто идет проверка на равенство полей объектов за счет перегруженных операторов.



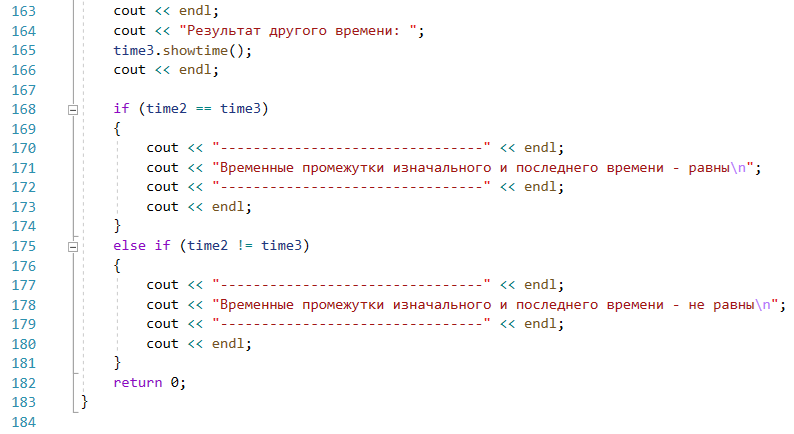
Код программы:



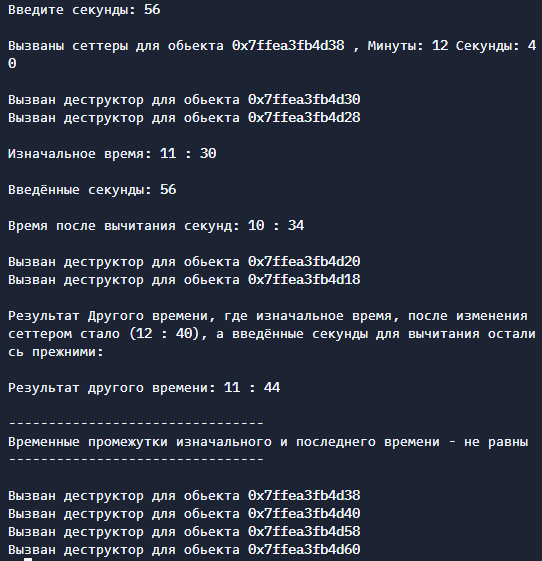








Результат программы:



**Контрольные вопросы:**

1. Для чего используются дружественные функции и классы?
2. Сформулировать правила описания и особенности дружественных функций.
3. Каким образом можно перегрузить унарные операции?
4. Сколько операндов должна иметь унарная функция-операция, определяемая внутри класса?
5. Сколько операндов должна иметь унарная функция-операция, определяемая вне класса?
6. Сколько операндов должна иметь бинарная функция-операция, определяемая внутри класса?
7. Сколько операндов должна иметь бинарная функция-операция, определяемая вне класса?
8. Чем отличается перегрузка префиксных и постфиксных унарных операций?
9. Каким образом можно перегрузить операцию присваивания?
10. Что должна возвращать операция присваивания?
11. Каким образом можно перегрузить операции ввода-вывода?

**Ответы:**

1. Дружественные функции применяются для доступа к скрытым полям класса и представляют собой альтернативу методам. Метод, как правило, описывает свойство объекта, а в виде дружественных функций оформляются действия, не являющиеся свойствами класса. То есть дружественный метод - это функция внутри класса, которая может обращаться к полям другого класса, а дружественный класс – это когда один класс, может работать с полями другого класса.
2. Дружественная функция объявляется внутри класса, к элементам которого ей нужен доступ, с ключевым словом friend. В качестве параметра ей должен передаваться объект или ссылка на объект класса, поскольку ключевой указатель this данной функции не передается. Место объявления дружественной функции в классе безразлично, на неё не действуют спецификаторы доступа. Дружественной функция не наследуется, также она может быть дружественной для других классов.
3. Унарные операции, перегружаемые в рамках определенного класса, могут перегружаться только через нестатическую компонентную функцию без параметров. Вызываемый объект класса автоматически воспринимается как операнд. Унарные операции, перегружаемые вне

области класса (как глобальные функции), должны иметь один параметр типа класса.

1. Унарная операция по определению работает с одним операндом. Раз она перегружается как компонентная нестатическая функция, то она не должна принимать параметров. (неявно принимает указатель *this*).
2. Т.к. оператор перегружается как глобальная функция, то параметр *this* ему не передается, следовательно, необходимо явно передавать объект класса.
3. Если оператор перегружается как компонентная функция, то левым операндом по умолчанию является объект класса – *this.* Правым операндом является тот объект, что передается в качестве параметра в перегружаемый оператор. (1)
4. Если оператор перегружается как не компонентная функция, чаще всего дружественная классу. То указатель *this* не передается, поэтому, необходимо явно передавать объект класса в качестве параметра. Также необходимо передавать объект другого класса, с которым должен взаимодействовать исходный класс посредством оператора. (2)
5. Префиксные и постфиксные операции по сути являются версией одного оператора в разных формах. Если при перегрузке префиксного оператора не нужно передавать никаких параметров, то при перегрузке постфиксного оператора необходимо передать незначащий параметр *int.* – Чтобы объяснить компилятору разницу.
6. Оператор присваивания можно перегрузить только как нестатическую компонентную функцию класса.
7. Ссылку на объект класса, в который происходит копирование (левый операнд). Это нужно для реализации многочисленного присваивания.
8. Для того, чтобы обеспечить взаимодействие пользовательского класса и потокового класса (левым операндом является объект потокового класса, правым операндом является объект пользовательского класса), необходимо перегрузить оператор << или оператор >> как дружественную функцию. С двумя параметрами – первый: объект класса std::ostream или std::istream, второй: объект пользовательского класса.
9. Как вызов метода класса: s.operator ++ ();
10. Как вызов глобальной функции: operator ++ (s);
11. Приоритет оператора << выше, чем у оператора <. Программа просто не скомпилируется.
12. Если выполнялась бы операция cout << (a<b); То компилятор это воспринял бы как вызов метода класса a.operator < (b)