Пермский национальный исследовательский политехнический университет.

Лабораторная работа № 3 по ООП.

«Перегрузка операций».

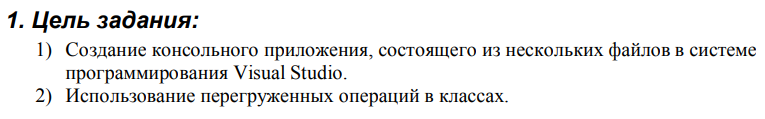
Выполнил: студент группы РИС-23-2б

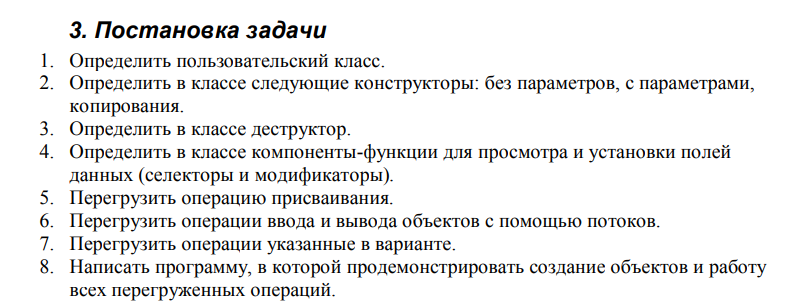
Вековшинин Иван Николаевич

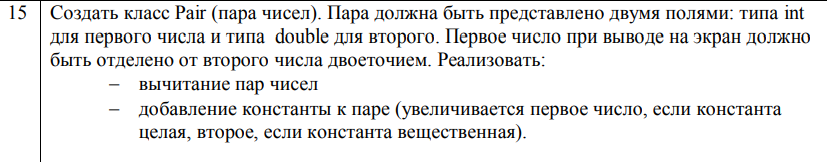
Проверила: доцент кафедры ИТАС

О.А. Полякова.

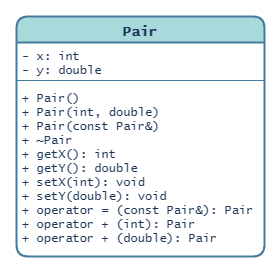
2024 г.



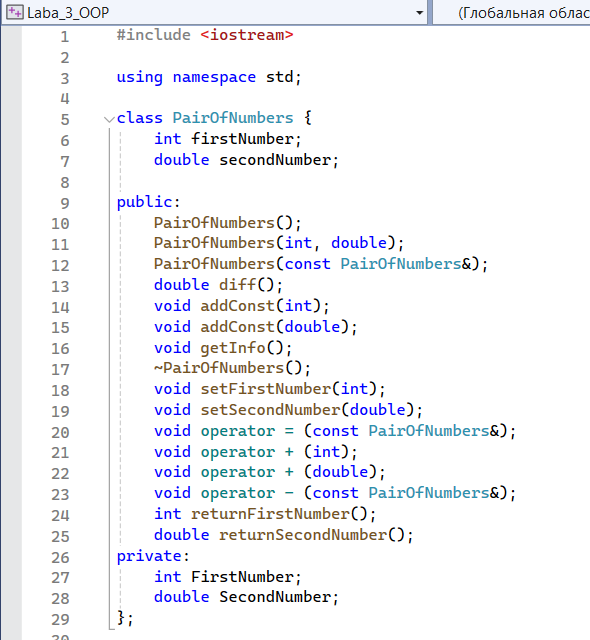




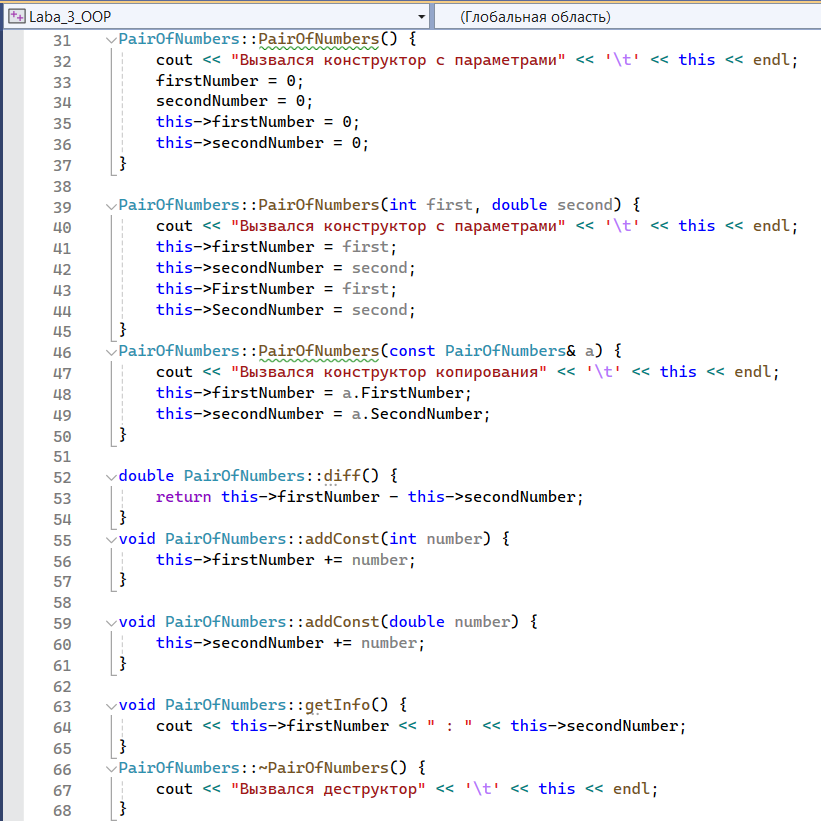
UML Diagram

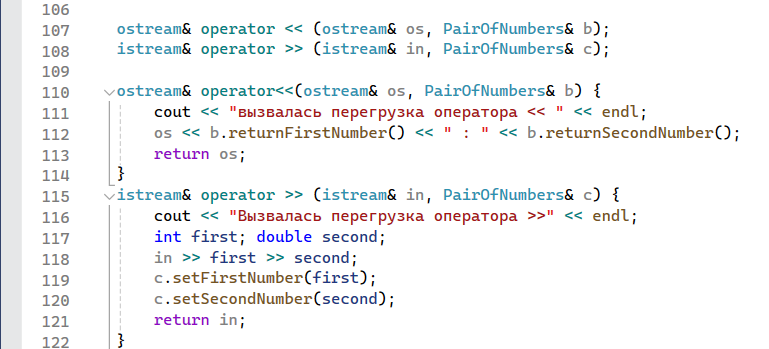


Class

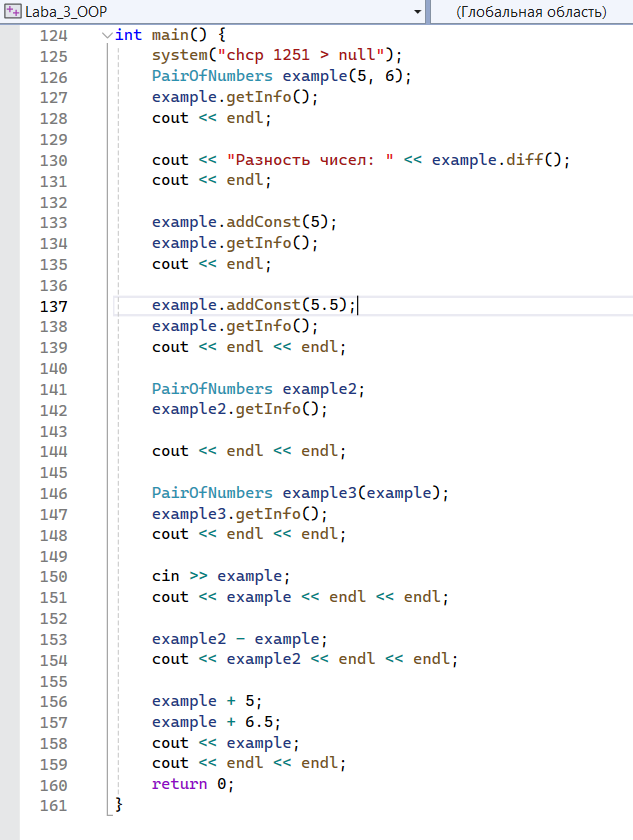


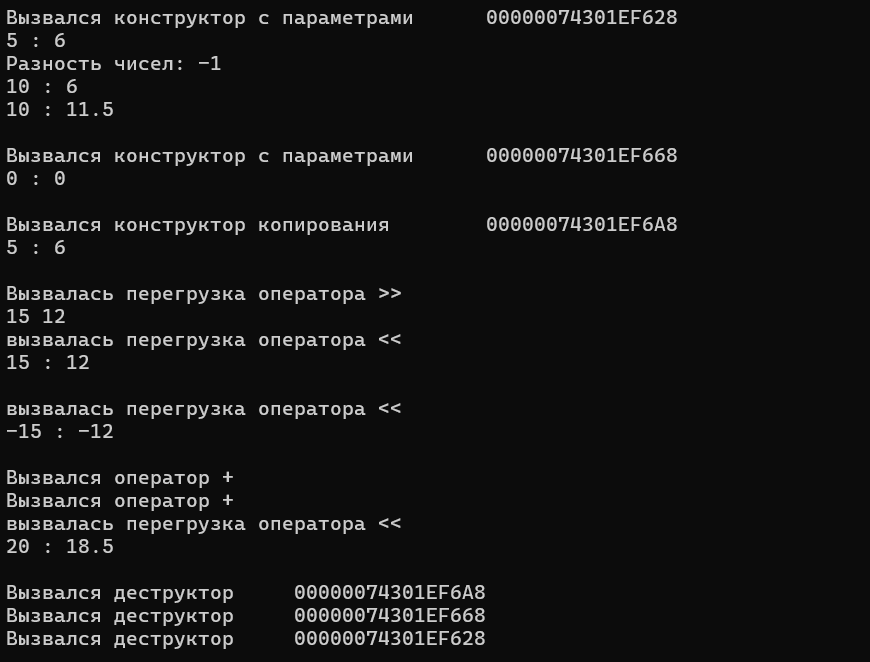
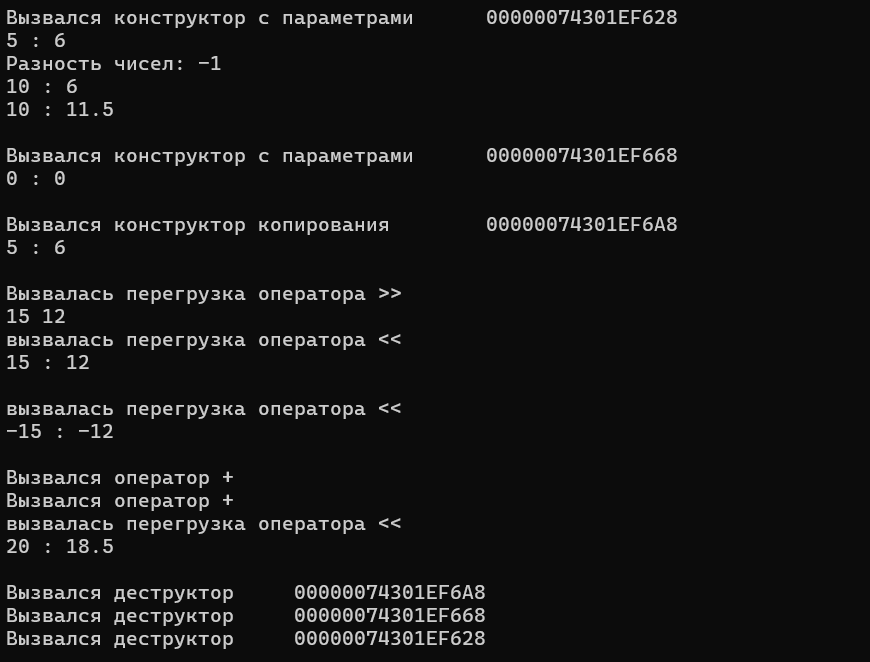
Function



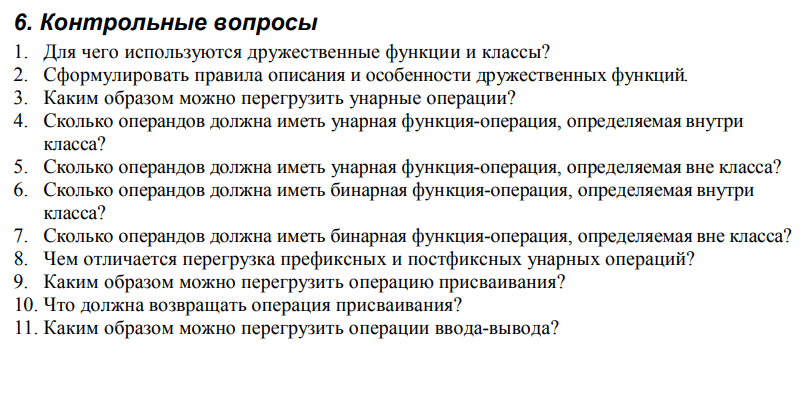


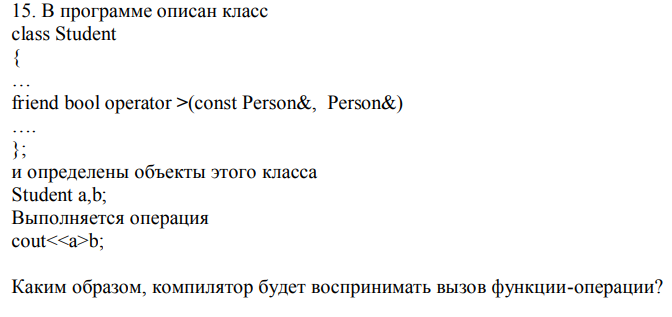
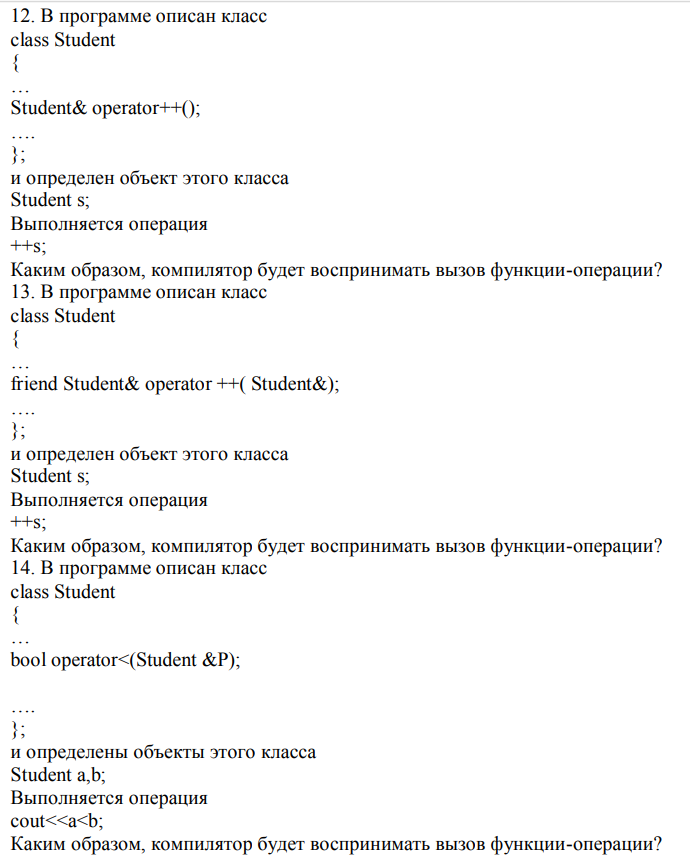






Ответы на вопросы:





1. Дружественные функции и классы используются для предоставления доступа к закрытым членам класса другим функциям или классам без необходимости делать их членами класса.

2. Правила описания и особенности дружественных функций:

- Дружественные функции объявляются внутри класса с использованием ключевого слова friend.

- Дружественная функция не является методом класса, но имеет доступ к закрытым членам класса.

- Дружественные функции могут быть определены как внутри, так и вне класса.

3. Унарные операции можно перегрузить с помощью методов класса, которые имеют только один операнд (обычно сам объект класса).

4. Унарная функция-операция, определяемая внутри класса, должна иметь один операнд - объект класса.

5. Унарная функция-операция, определяемая вне класса, также должна иметь один операнд - объект класса.

6. Бинарная функция-операция, определяемая внутри класса, должна иметь два операнда - обычно сам объект класса и второй объект.

7. Бинарная функция-операция, определяемая вне класса, также должна иметь два операнда - два объекта или объект и другой тип данных.

8. Перегрузка префиксных и постфиксных унарных операций отличается тем, что префиксная операция выполняется до изменения значения объекта, а постфиксная - после изменения.

9. Операцию присваивания можно перегрузить с помощью метода operator= внутри класса.

10. Операция присваивания должна возвращать ссылку на объект для поддержки цепочечных присваиваний.

11. Операции ввода-вывода можно перегрузить с помощью методов operator<< и operator>> для вывода и ввода объектов вашего класса через потоки std::cout и std::cin.

12. Когда операция ++s выполняется для объекта s, компилятор будет искать метод operator++() в классе Student. Поскольку этот метод возвращает ссылку на объект Student, операция ++s будет выполняться на самом объекте s, увеличивая его значение в соответствии с логикой, определенной в методе operator++().

13. В данном случае, когда операция ++s выполняется для объекта s, компилятор будет искать дружественную функцию operator++( Student&). Поскольку эта функция является дружественной для класса Student, она имеет доступ ко всем закрытым членам класса и может изменять его значения. Таким образом, операция ++s будет вызывать эту дружественную функцию для увеличения значения объекта s в соответствии с логикой, определенной в функции operator++().

14. При вызове оператора < для объектов a и b, компилятор будет искать метод operator<() в классе Student. Поскольку этот метод принимает ссылку на объект Student, операция a < b будет вызывать этот метод для объекта a, передавая объект b в качестве аргумента. Результат выполнения этого метода будет использоваться для вывода результата с помощью cout.

15. При вызове оператора > для объектов a и b, компилятор будет искать дружественную функцию operator>() вне класса Student. Поскольку эта функция является дружественной для класса Student, она имеет доступ ко всем закрытым членам класса. Операция a > b будет вызывать эту дружественную функцию, передавая объекты a и b в качестве аргументов. Результат выполнения этой функции будет использоваться для вывода результата с помощью cout.