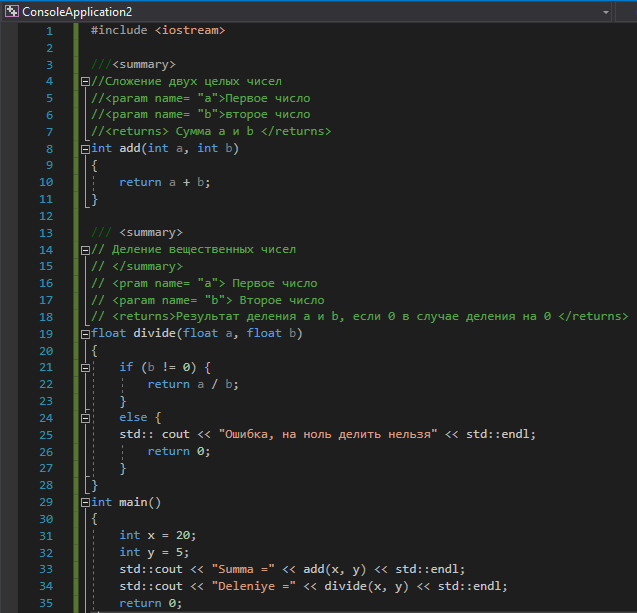
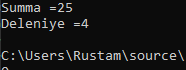
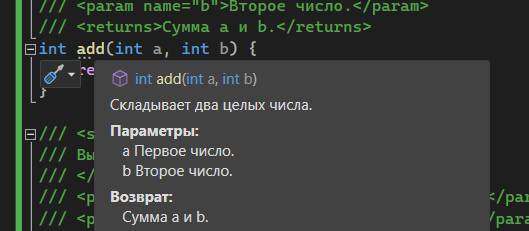
**Обеспечение качества кода**

**Самодокументируемый код**

**Отчет**

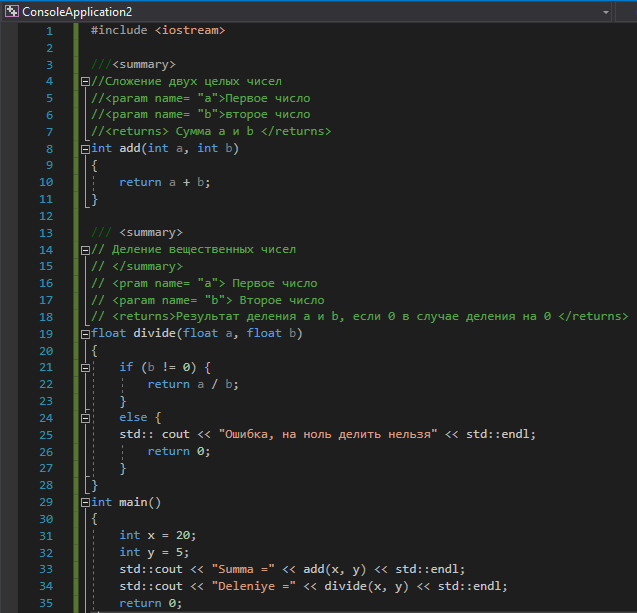






**Соглашение о кодировании**

1. По возможности используйте современные языковые функции и версии C#.
2. Избегайте устаревших или устаревших конструкций языка.
3. Перехват исключений, которые можно правильно обрабатывать; избегайте перехвата универсальных исключений.
4. Используйте определенные типы исключений для предоставления значимых сообщений об ошибках.
5. Используйте запросы и методы LINQ для обработки коллекций, чтобы улучшить удобочитаемость кода.
6. Используйте асинхронное программирование с асинхронным и ожиданием операций с привязкой ввода-вывода.
7. Будьте осторожны с взаимоблокировками и используйте [Task.ConfigureAwait](https://learn.microsoft.com/ru-ru/dotnet/api/system.threading.tasks.task.configureawait) при необходимости.
8. Используйте языковые ключевое слово для типов данных вместо типов среды выполнения. Например, используйте string вместо [System.String](https://learn.microsoft.com/ru-ru/dotnet/api/system.string) нее или int вместо [System.Int32](https://learn.microsoft.com/ru-ru/dotnet/api/system.int32).
9. Используйте int вместо неподписанных типов. Использование int часто используется на C#, и при использовании int проще взаимодействовать с другими библиотеками. Исключения предназначены для документации, конкретной для типов данных без знака.
10. Используйте var только в том случае, если средство чтения может вывести тип из выражения. Читатели просматривают наши примеры на платформе документов. У них нет подсказок по наведении указателя мыши или инструментов, отображающих тип переменных.
11. Напишите код с четкостью и простотой.
12. Избегайте чрезмерно сложной и запутанной логики кода.



**Анализ качества кода**

Логическая сложность программы Джилб определяет как насыщенность программы условными операторами типа IF-THEN-ELSE и операторами цикла (при этом следует учитывать, что фактическая запись условий и циклов в разных языках программирования может быть представлена в разной форме при сохранении указанного смысла операторов). При этом вводятся следующие характеристики программного средства:

• CL - абсолютная сложность программы, характеризуемая количеством операторов условий;

• cl - относительная сложность программы, определяющая насыщенность программы операторами условия

CL =1

N=13

cl = 0,07

Метрика Джилба - это метрика, которая используется для оценки сложности программы. Она основывается на количестве строк кода и количестве операторов ветвления и циклов.