**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**КАФЕДРА САПР**

**ОТЧЕТ**

**по лабораторной работе №4**

**по дисциплине «Объектно-ориентированное программирование»**

**Тема: Использование методов**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 9301 |  | Синицкая В. А. |
| Преподаватель |  | Новакова Н.Е. |

Санкт-Петербург

2021

Оглавление

[1 Цель работы 3](#_Toc65267574)

[2 Анализ задачи 3](#_Toc65267575)

[3 Формальная постановка задачи 3](#_Toc65267576)

[3.1 Исходные данные 3](#_Toc65267577)

[3.2 Результирующие (выходные) данные 3](#_Toc65267578)

[4 Используемые классы и методы 4](#_Toc65267579)

[5 Текст программы 4](#_Toc65267580)

[6 Контрольный пример 7](#_Toc65267581)

[7 Полученные результаты 8](#_Toc65267582)

[8 Выводы 8](#_Toc65267583)

[9 Список использованных источников 9](#_Toc65267584)

# Цель работы

Получение знаний и навыков в использовании классов и их методов. Создание методов с разными типами параметров и возвращаемых значений, практика в работе с конструкцией try-catch, осуществление вызова public static методов одного класса в другом классе.

# Анализ задачи

Требуется:

1. Написать программу, принимающую два целочисленных значения и возвращающую большее из них.
2. Написать программу, не возвращающую значений, но меняющая местами значения переданных по ссылке параметров.
3. Написать метод Factorial, имеющий один входной и один выходной параметр, возвращающий правду или ложь, в зависимости от того, пришлось ли методу обработать исключение (были ли данные введены неверно).
4. Написать рекурсивный метод вычисления факториала RecursiveFactorial.

# Формальная постановка задачи

## Исходные данные

В качестве исходных данных выступают два целых числа для сравнения и обмена значениями, далее число для вычисления факториала методом Factorial и число для вычисления факториала методом RecursiveFactorial.

## Результирующие (выходные) данные

Пользователю в консоль по ходу выполнения программы транслируются подсказки, что делать. Результат работы для первых двух чисел: 2 числа, напечатанные пользователем, обмененные местами 2 этих числа, большее из них. Для последующих двух чисел выводятся их факториалы.

# Используемые классы и методы

В программах используются методы Console.WriteLine() и Console.ReadLine() - методы класса System.Console. System.Console – это класс для работы с консольным окном, определенный разработчиками стандартных библиотек для языка C#.

Метод Console.ReadLine() сохраняет введенную пользователем строку в заданную переменную.

Метод Console.WriteLine() служит для отображения в консольном окне пользователя строк и других данных, переданных в данный метод в качестве параметров.

Программа содержит два класса: class Utils и class Test. В первом содержатся методы Greater, Swap, Factorial, RecursiveFactorial. Во втором содержится метод Main, в котором осуществляются вызовы методов класса Utils посредством точечной нотации. Метод Greater принимает 2 целых числа и возвращает большее, метод Swap не возвращает ничего, но меняет местами значения двух переданных ему по ссылке параметров, метод Factorial принимает входной параметр, обозначающий число, факториал которого нужно вычислить, выходной параметр, который будет изменен после выполнения метода подобно ссылочному, метод возвращает bool-значение, правда соответствует безошибочной работе, а ложь — прерванному выполнению вследствие обнаружения ошибки во входных данных. Метод recursiveFactorial работает подобно методу Factorial, однако отличается тем, что является рекурсивной версией, то есть для вычисления результата вызывает сам себя, имея крайние случаи, в которых ответ известен сразу же.

# Текст программы

1. using System;
3. namespace OOP4LabaUtils
4. {
5. class Utils
6. {
7. public static int Greater(int a, int b)
8. { return (a > b ? a : b); } // if a > b returns a, returns b otherwise
10. public static void Swap(ref int a, ref int b)
11. {
12. int temp = b;
13. b = a;
14. a = temp;
15. }
16. public static bool Factorial(in int n, out int answer)
17. {
18. answer = 1;
19. try
20. {
21. if (n < 0)
22. throw new ArgumentOutOfRangeException("factorial can't be taken from a negative number");
23. for (int i = 2; i <= n; i++)
24. answer \*= i;
25. return true;
26. }
27. catch (ArgumentOutOfRangeException negative\_number)
28. {
29. Console.WriteLine("Error: {0} \nStack: {1}", negative\_number.Message, negative\_number.StackTrace);
30. return false;
31. }
32. }
34. public static bool RecursiveFactorial(in int n, out int answer)
35. {
36. answer = n;
37. try
38. {
39. if (n < 0)
40. throw new ArgumentOutOfRangeException("factorial can't be taken from a negative number");
41. else if (n <= 1)
42. answer = 1;
43. else
44. {
45. int sub\_answer;
46. if (RecursiveFactorial(n - 1, out sub\_answer))
47. answer \*= sub\_answer;
48. }
49. return true; // true if n = 0 and if n = 1
50. }
51. catch (ArgumentOutOfRangeException negative\_number)
52. {
53. Console.WriteLine("Error: {0} \nStack: {1}", negative\_number.Message, negative\_number.StackTrace);
54. return false;
55. }
56. }
57. }
59. class Test
60. {
61. static void Main(string[] args)
62. {
63. int x, y, greater;
64. Console.WriteLine("Please, enter first integer number for comparison");
65. x = int.Parse(Console.ReadLine());
66. Console.WriteLine("Please, enter second integer number for comparison");
67. y = int.Parse(Console.ReadLine());
68. Console.WriteLine("You've entered numbers: {0}, {1}", x, y);
69. Utils.Swap(ref x, ref y);
70. Console.WriteLine("The result of swap: {0}, {1}", x, y);
71. greater = Utils.Greater(x, y);
72. Console.WriteLine("The greater number is: {0}", greater);
73. Console.WriteLine("Please, enter the integer positive number for calculation the factorial");
74. x = int.Parse(Console.ReadLine());
75. if (Utils.Factorial(x, out y))
76. Console.WriteLine(" {0}! = {1}", x, y);
77. Console.WriteLine("Please, enter the integer positive number for calculation the factorial");
78. x = int.Parse(Console.ReadLine());
79. if (Utils.RecursiveFactorial(x, out y))
80. Console.WriteLine(" {0}! = {1}", x, y);
81. }
82. }
84. }

# Контрольный пример

На Рис. 6.1 представлен результат работы при сравнении и обмене двух положительных чисел и при положительных числах для вычисления факториала.

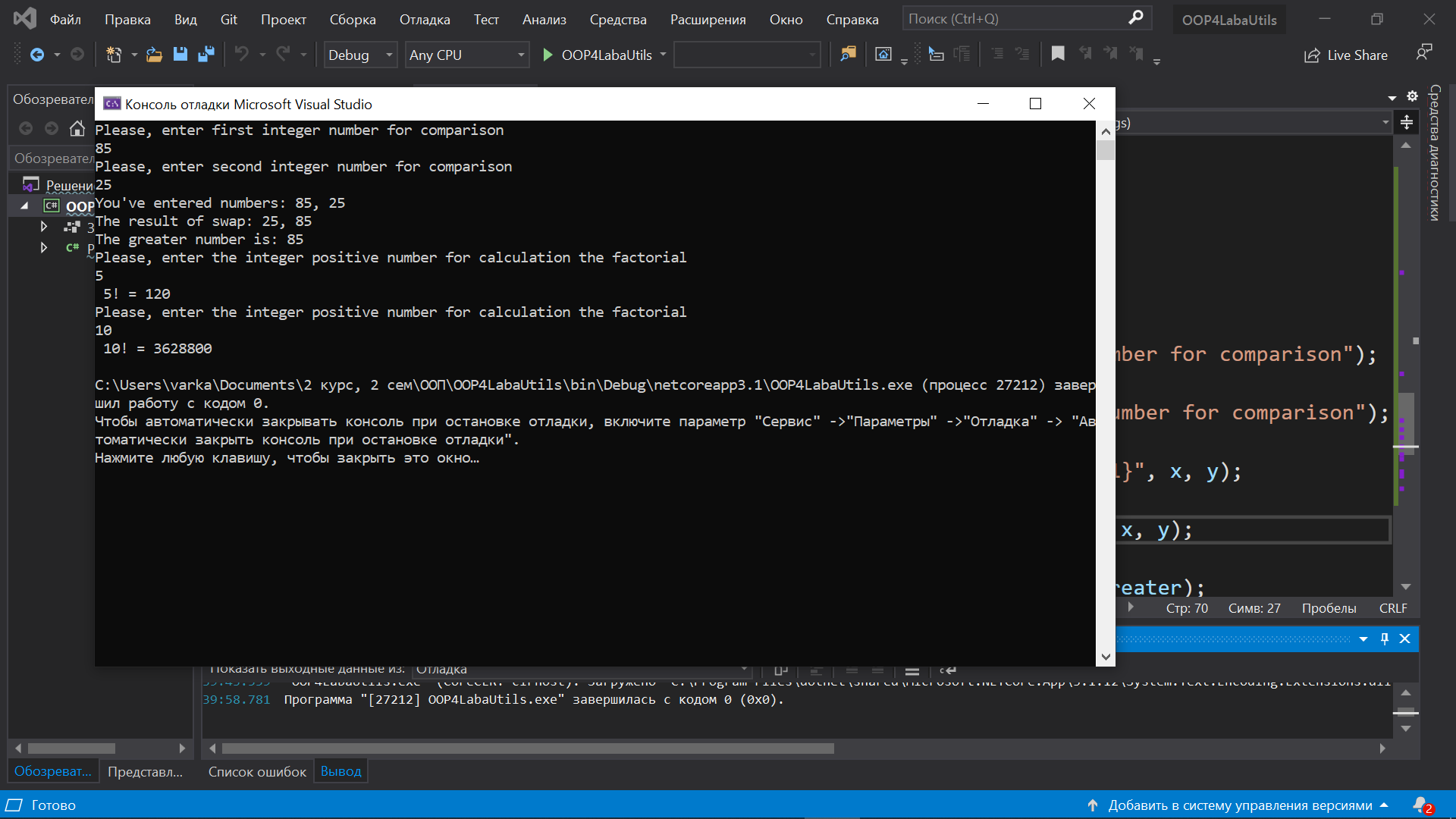


Рис. . — пример работы программы при корректных данных.

На Рис. 6.2 представлена работа программы, показывающая вычисление 1! и 0!.

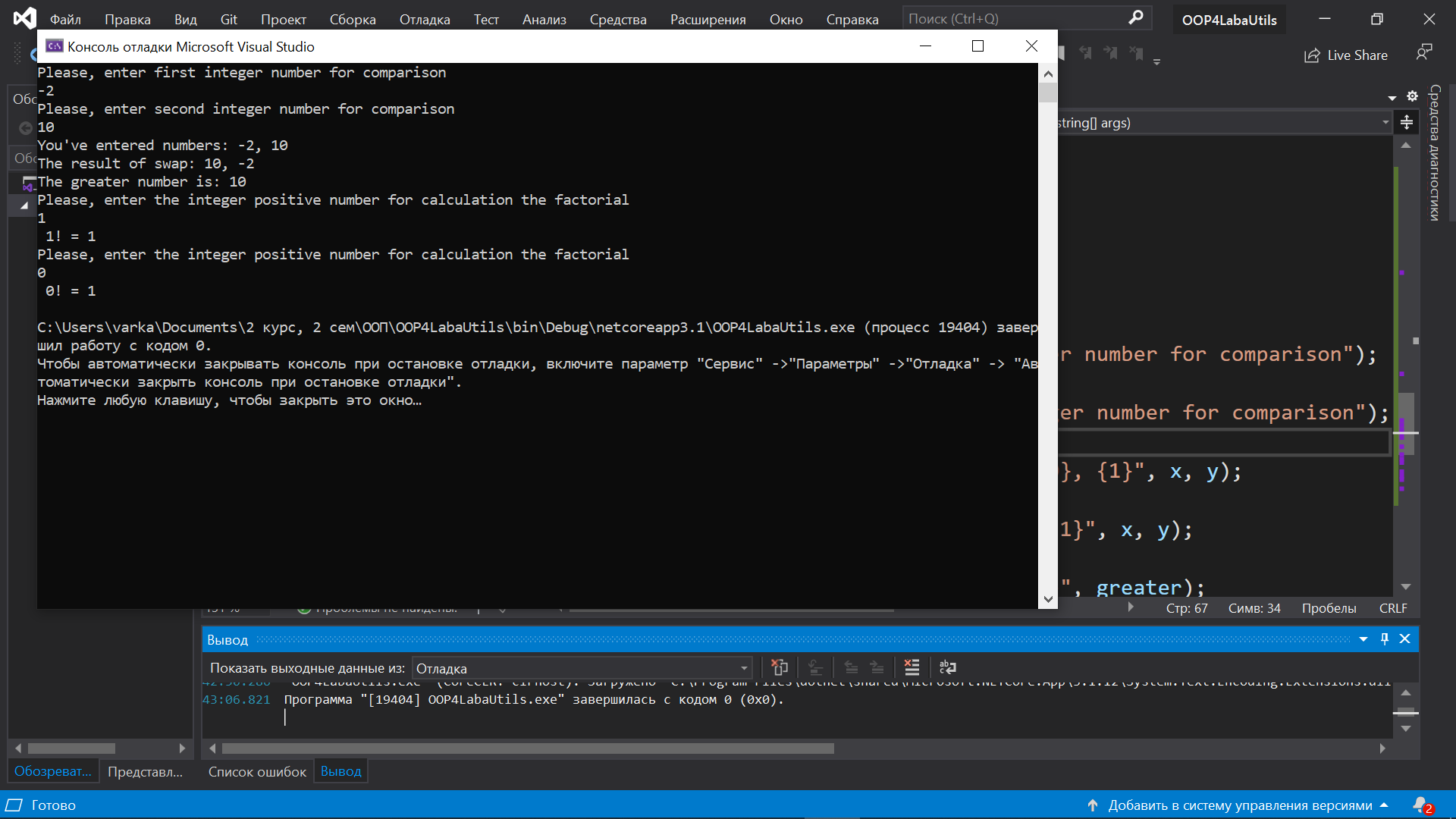


Рис. . — работа программы при сравнении отрицательного и положительного числа, вычисление 1! и 0!

На Рис. 6.3 показана работа программы при сравнении отрицательных чисел и при вызове методов вычисления программы при отрицательных числах, что вызывает исключение, обработку которого пользователь видит в виде пояснительного сообщения о сути проблемы и ее местоположении.

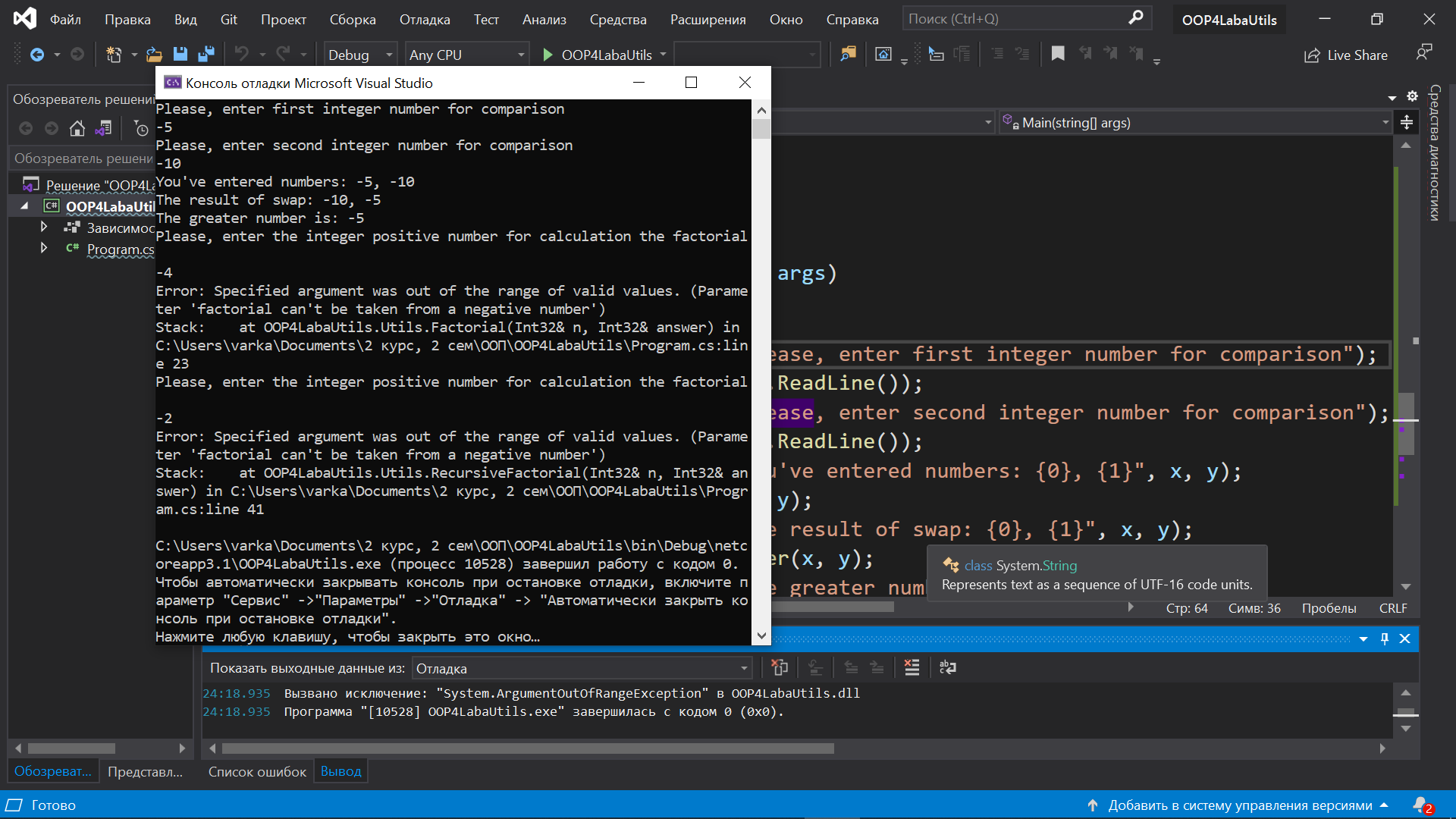


Рис. . — работа программы при отрицательных числах.

# Полученные результаты

В результате работы программы пользователю выводится большее из двух чисел и результаты вычисления факториалов для введенных чисел. При отрицательных числах для факториалов пользователь видит пояснительное сообщение. Ошибка в одном из методов не влечет за собой аварийного завершения программы, ее выполнение продолжится, если после нее еще есть операции.

# Выводы

В ходе выполнения данной лабораторной работы:

1. Были получены навыки использования классов и их методов.
2. Были получены знания и навыки написания программ на C#, использующих try-catch и различные спецификаторы передаваемых в метод параметров.
3. Практиковалась работа с преобразованием типа данных и взаимодействие с консолью.
4. Был получен опыт написания рекурсивной функции на C#.

# Список использованных источников

Руководство по языку C#: URL: <https://docs.microsoft.com/ru-ru/dotnet/csharp/> (дата обращения: 26.02.2021)

Материалы учебного курса по ООП: URL: <https://vec.etu.ru/moodle/course/view.php?id=5512> (дата обращения: 26.02.2021)