Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Колледж информатики и программирования

Специальность 09.02.03 «Программирование в компьютерных системах»

**ОТЧЕТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ**

Профессиональный модуль ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

*(наименование профессионального модуля)*

Выполнил:

обучающийся учебной группы № 420

М.Р. Осколков

*(И.О. Фамилия)*

Проверил:

руководитель практики от колледжа:

И.В. Сибирев

*(И.О. Фамилия)*

Е.Л. Альшакова

*(И.О. Фамилия)*

**Москва**

**20­­­­22**

**Перечень заданий/работ, выполненных в ходе учебной практики**

Специальность 09.02.03 «Программирование в компьютерных системах»

*(наименование специальности)*

Профессиональный модуль ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

*(наименование профессионального модуля)*

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Темы учебной практики** | **Выполненные задания/работы** |
| 1 | Тема 1. Программирование разветвляющегося процесса | Разработать программу сравнения двух чисел соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 2 | Тема 2. Программирование разветвляющихся алгоритмов с помощью команд передачи управления языка Assembler | Реализовать алгоритм ветвления на языке Assembler соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 3 | Тема 3. Работа с флагами регистра флагов с использованием команд языка Assembler | Разработать программу с использованием регистра FLAGS соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 4 | Тема 4. Программирование алгоритмов с использованием арифметических команд и флага переноса языка Assembler | Разработать программу решения квадратного уравнения соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 5 | Тема 5. Выполнение операций сдвига и логических операций над числами с помощью команд языка Assembler | Разработать программу с логических преобразований чисел соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 6 | Тема 6. Умножение знаковых чисел | Разработать программу вычисления произведений произвольных чисел соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 7 | Тема 7. Работа с флагами CF, OF. Деление знаковых чисел | Разработать программу вычисления деления произвольных чисел соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 8 | Тема 8. Циклы | Разработать программу вычисления значений функций разложением в ряд Тейлора соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 9 | Тема 9. Работа с сегментами. Подпрограммы | Разработать программу с использованием подпрограмм соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 10 | Тема 10. Обработка одномерных массивов с использованием команд языка Assembler | Разработать программу обработки одномерного массива соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 11 | Тема 11. Работа со стеком | Разработать программу с использованием стека соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 12 | Тема 12. Работа с файлами | Разработать программу работы с файлом соблюдая следующие этапы:  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 13 | Тема 13. Ассемблерная вставка в С++ | Разработать программу на языке соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 14 | Тема 14. Разработка программы с использованием Windows Forms | Разработать программу с использованием Windows Forms соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 15 | Тема 15. Написание программы для произведения стереометрических расчетов | Разработать программу для произведения стереометрических расчетов соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 16 | Тема 16. Написание консольного приложения для вывода числового массива | Разработать консольное приложение для вывода числового массива соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 17 | Тема 17. Написание приложения для работы с символьным массивом | Разработать приложение для работы с символьным массивом соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 18 | Тема 18. Создание приложения с использованием компонента ListBox | Разработать приложение с использованием компонента ListBox соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 19 | Тема 19. Создание приложения с использованием компонента DataGridView и многострочным textBox | Разработать приложение с использованием компонента DataGridView и многострочным textBox соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм;  - написать программный код в соответствии с разработанными алгоритмами и проектом;  - выполнить отладку;  - разработать техническую документацию. |
| 20 | Тема 20. Создание приложения с для произведения планиметрических расчетов с использованием классов | Разработать приложение с для произведения планиметрических расчетов с использованием классов соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 21 | Тема 21. Рефакторинг и оптимизация ранее разработанного приложения для обработки числового массива с использованием классов | Произвести разработку приложения для обработки числового массива с использованием классов соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 22 | Тема 22. Создание приложения с использованием свойства и методов класса String | Разработать приложение с использованием свойств и методов класса String соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 23 | Тема 23. Создание приложения с использованием делегатов (часть 1) | Разработать приложение с использованием делегатов соблюдая следующие этапы: - определить спецификации;  - создать алгоритм. |
| 24 | Тема 24. Создание приложения с использованием делегатов (часть 2) | Разработать приложение с использованием делегатов соблюдая следующие этапы:  - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 25 | Тема 25. Создание приложения с использованием событий | Разработать приложение с использованием событий соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 26 | Тема 26. Создание приложения «Калькулятор» с использованием классов | Разработать приложение «Калькулятор» с использованием событий соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 27 | Тема 27. Создание приложения для проверки орфографии | Разработать приложение для проверки орфографии соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 28 | Тема 28. Создание приложения для организации взаимодействия с MS Word | Разработать приложение для организации взаимодействия с MS Word соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 29 | Тема 29. Создание приложения с использованием функций MS Excel | Разработать приложение с использованием функций MS Excel соблюдая следующие этапы:  - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 30 | Тема 30. Создание поточного приложения | Разработать поточное приложение соблюдая следующие этапы: - определить спецификации;  - создать алгоритм; - при необходимости провести проектирование; - написать программный код в соответствии с разработанными алгоритмами и проектом; - выполнить отладку; - разработать техническую документацию. |
| 31 | Тема 31. Защита отчета по практике | Защита сформированного отчета по практике. |

**СОДЕРЖАНИЕ**

**Задание 1.**

Разместить в памяти компьютера массив из N целых беззнаковых чисел. Написать программу, позволяющую найти сумму всех элементов массива, больше 10.

format PE console

include 'win32a.inc'

entry start

proc arifmet

mov esi,mas

mov ecx,10

mov eax,0

mov edx,10

me:

cmp edx,[esi]

jl me1

jmp m

me1:

add eax,[esi]

m:

add esi,4

loop me

ret

endp

start:

macro prntn massiv, N

{

local metka

mov esi,massiv

mov ecx, [N]

metka:

mov eax,[esi]

push ecx

cinvoke printf,fmat,eax

pop ecx

add esi,4

loop metka

}

macro input N, A

{

local metka

mov esi, A

mov ecx, [N]

cycl:

push ecx

cinvoke printf,text4

cinvoke scanf, fmat, K

pop ecx

mov ebx, [K]

mov [esi], ebx

add esi, 4

loop cycl

}

cinvoke printf,text2

cinvoke scanf, fmat, N

input N,mas

cinvoke printf,text3

prntn mas,N

call arifmet

cinvoke printf,text1,eax

cinvoke system,p

invoke exit,0

invoke sleep,5000

text1 db 10,13,'The amount: %d', 10,13

text2 db 10,13,'Enter N: ', 0

text3 db 10,13,'Array: ', 0

text4 db 10,13,'Enter a number from massiv: ', 0

fmat db ' %d',0

p db 'pause', 0

K dd ?

N dd ?

mas dd N dup(?)

data import

library msvcrt, 'MSVCRT.DLL',\

kernel132, 'KERNEL32.DLL'

import kernel132,\

sleep, 'Sleep'

import msvcrt,\

printf, 'printf',\

scanf,'scanf',\

exit, 'exit',\

setlocal, 'setlocal',\

system, 'system'

end data

Примеры работы кода

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описание

**Задание 2.**

Сформировать одномерный массив a(10) из произвольных чисел, принадлежащих отрезку [-2, 6] и вычислить среднее арифметическое тех его элементов, значения которых не превышают заданного числа z. Заменить отрицательные элементы массива найденным средним арифметическим. Программу разработать на основе пользовательских функций.

format PE console

include 'win32a.inc'

entry start

proc average

push ebp

mov ebp, esp

push ebx

push ecx

mov ecx, [ebp+8]

mov ebx, [ebp+12]

xor eax,eax

xor edi,edi

jcxz m4

mov esi,[z]

cycln:

cmp [ebx],esi

jg m5

inc edi

add eax,[ebx]

m5:

add ebx,4

loop cycln

cdq

idiv edi

m4:

pop ecx

pop ebx

pop ebp

mov [arr], eax

ret 8

endp

proc save

push ebp

mov ebp, esp

push ebx

push ecx

mov edx, [ebp+8]

mov ecx, [ebp+12]

mov ebx, [ebp+16]

jcxz m1

sev:

cmp [ebx], dword 0

jge m2

mov eax, edx

mov [ebx], eax

m2:

add ebx, 4

loop sev

m1:

pop ecx

pop ebx

pop ebp

ret 12

endp

start:

macro prntn massiv, N

{

local metka

mov esi,massiv

mov ecx,[N]

metka:

mov eax,[esi]

push ecx

cinvoke printf,fmat,eax

pop ecx

add esi,4

loop metka

}

cinvoke printf,text3

prntn a, N

lea eax,[a]

push eax

mov eax,[N]

push eax

call average

push eax

cinvoke printf,text1,[arr]

pop eax

mov ebx,eax

lea eax,[a]

push eax

mov eax,[N]

push eax

push ebx

call save

cinvoke printf,text3

prntn a, N

invoke sleep,5000

text1 db 10,13,'answer : %d', 10,13

text3 db 10,13,'Array: ', 0

fmat db ' %d',0

p db 'pause', 0

a dd 6,-2,-1,-1,6,6,5,6,6,-1

N dd 10

z dd 6

arr dd ?

mas dd N dup(?)

data import

library msvcrt, 'MSVCRT.DLL',\

kernel132, 'KERNEL32.DLL'

import kernel132,\

sleep, 'Sleep'

import msvcrt,\

printf, 'printf',\

scanf,'scanf',\

exit, 'exit',\

setlocal, 'setlocal',\

system, 'system'

end data

Пример выполнения кода:

Изображение выглядит как текст

Автоматически созданное описание

**Задание 3.**

Описать рекурсивную булевскую функцию equal(N, S) (где N≥0 и S –целые числа), которая проверяет, совпадает ли сумма цифр десятичной записи числа N со значением S. Например: equal(12345,15)=true, equal(24,7)=false, equal(100,1)=true

format PE console

include 'win32a.inc'

entry start

proc equal

mov edx, 0

mov eax, [a]

cmp eax, 0

jg f2

ret

f2:

idiv ecx

mov [a], eax

mov ebx, [c]

add ebx, edx

mov [c], ebx

call equal

ret

endp

start:

cinvoke printf,text1

cinvoke scanf, fmat, N

mov eax, [N]

mov [a], eax

cinvoke printf,text3

cinvoke scanf, fmat, S

mov ebx, 0

mov ecx, 10

call equal

mov eax, [S]

mov ebx, [c]

cmp ebx, eax

je met

cinvoke printf,text4,[N],[S]

cinvoke printf,text6

jmp end\_

met:

cinvoke printf,text4,[N],[S]

cinvoke printf,text5

end\_:

invoke sleep,5000

text1 db 10,13,'Enter N: ', 0

text2 db 10,13,' %d ', 0

text3 db 10,13,'Enter S: ', 0

text4 db 10,13,'equal(%d, %d) = ', 0

text5 db 'true', 0

text6 db 'false', 0

fmat db ' %d',0

p db 'pause', 0

N dd ?

S dd ?

a dd ?

c dd ?

data import

library msvcrt, 'MSVCRT.DLL',\

kernel132, 'KERNEL32.DLL'

import kernel132,\

sleep, 'Sleep'

import msvcrt,\

printf, 'printf',\

scanf,'scanf',\

exit, 'exit',\

setlocal, 'setlocal',\

system, 'system'

end data

Пример выполнения кода:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст, монитор, экран, снимок экрана

Автоматически созданное описание

**Задание 4.**

Заменить элементы модуль которых в остатке 2 на нули, найти ID максимального походящего числа

format PE console

include 'win32a.inc'

entry start

proc maxim

lea ebx,[X]

mov ecx,[N]

mov edx,[N]

inc edx

mov eax,0

mov [ID],ecx

cycl:

cmp eax,[ebx]

jg cycl1

mov [ID],edx

sub [ID],ecx

mov eax,[ebx]

cycl1:

add ebx,4

loop cycl

ret

endp

proc zamena

mov esi,X

mov ecx,[N]

mov edx,0

metka:

mov eax,[esi]

getabs:

neg eax ;находим модуль числа

js getabs

mov ebx,5

idiv ebx

mov ebx, 2

cmp ebx,edx

je metka1

jmp met

metka1:

mov eax, 0

mov [esi],eax

met:

mov edx,0

add esi,4

loop metka

ret

endp

start:

;;;;;;Обнулить отрицательные элементы

macro prntn massiv, N

{

local metka

mov esi,massiv

mov ecx,[N]

metka:

mov eax,[esi]

push ecx

cinvoke printf,fmat,eax

pop ecx

add esi,4

loop metka

}

cinvoke printf,text2

prntn X, N

call zamena

cinvoke printf,text2

prntn X, N

call maxim

cinvoke printf,text1,eax

cinvoke printf,text3,[ID]

invoke sleep,5000

text1 db 10, 13, 'Max: %d', 0

text2 db 10,13,'Array: ', 0

text3 db 10, 13, 'Max elements ID: %d', 0

fmat db ' %d',0

p db 'pause', 0

ID dd ?

N dd 10

X dd 7,6,15,17,12,-12,4,0,-10,-22

data import

library msvcrt, 'MSVCRT.DLL',\

kernel132, 'KERNEL32.DLL'

import kernel132,\

sleep, 'Sleep'

import msvcrt,\

printf, 'printf',\

scanf,'scanf',\

exit, 'exit',\

setlocal, 'setlocal',\

system, 'system'

end data

пример выполнения кода:

Изображение выглядит как текст

Автоматически созданное описание

**Задание 5.**

Разместить в памяти компьютера матрицу 4\*3 целых чисел со знаком. Написать программу, позволяющую найти минимальный элемент среди элементов массива по каждой строке. Исходную матрицу и полученные значения вывести на экран.

proc minim

mov ecx,[lines]

m:

push ecx

lea ebx, [mas]

mov ecx,[columns]

cycl:

cmp ebx,[esi]

jl metka

mov ebx,[esi]

metka:

mov [min], ebx

add esi,4

loop cycl

cinvoke printf,text2,[counter],[min]

inc [counter]

pop ecx

loop m

ret

endp

macro prntn massiv,lines,columns

{

local metka

local met

mov esi,massiv

mov ecx,[lines]

met:

push ecx

mov ecx,[columns]

metka:

mov eax,[esi]

push ecx

cinvoke printf,fmat,eax

pop ecx

add esi,4

loop metka

cinvoke printf,text3

pop ecx

loop met

}

cinvoke printf,text1,[lines],[columns]

prntn mas,lines,columns

mov esi,mas

call minim

Пример входных и исходный данных:

Изображение выглядит как текст

Автоматически созданное описание Изображение выглядит как текст

Автоматически созданное описание

**Задание 6.**

|  |
| --- |
| Дана целочисленная прямоугольная матрица.  2) Найти номер первой из строк, не содержащих ни одного положительного элемента. |

format PE console

include 'win32a.inc'

entry start

proc poloz

mov ecx,[lines]

m:

push ecx

mov eax, 0

mov ebx, 0

mov ecx,[columns]

cycl:

cmp [esi],eax

jge metka

inc ebx

metka:

add esi,4

loop cycl

cmp ebx,[columns]

jne next

mov ebx, 0

cmp [s], ebx

jne next

mov ebx,[counter]

mov [s],ebx

next:

inc [counter]

pop ecx

loop m

cinvoke printf,text2,[s]

ret

endp

start:

macro prntn massiv,lines,columns

{

local metka

local met

mov esi,massiv

mov ecx,[lines]

met:

push ecx

mov ecx,[columns]

metka:

mov eax,[esi]

push ecx

cinvoke printf,fmat,eax

pop ecx

add esi,4

loop metka

cinvoke printf,text3

pop ecx

loop met

}

cinvoke printf,text1,[lines],[columns]

prntn mas,lines,columns

mov esi,mas

call poloz

invoke sleep,9000

text1 db 'Massiv (%d,%d):', 13,10,10,0

text2 db 10,13,'The number of the first line without positive elements = %d ',10,0

text3 db 10,13,'', 0

fmat db ' %d',0

p db 'pause', 0

lines dd 3

s dd 0

columns dd 4

counter dd 1

mas dd 12,-1,-3,-23,-4,-90,-4,-12,-1,-7,-2,-10

data import

library msvcrt, 'MSVCRT.DLL',\

kernel132, 'KERNEL32.DLL'

import kernel132,\

sleep, 'Sleep'

import msvcrt,\

printf, 'printf',\

scanf,'scanf',\

exit, 'exit',\

setlocal, 'setlocal',\

system, 'system'

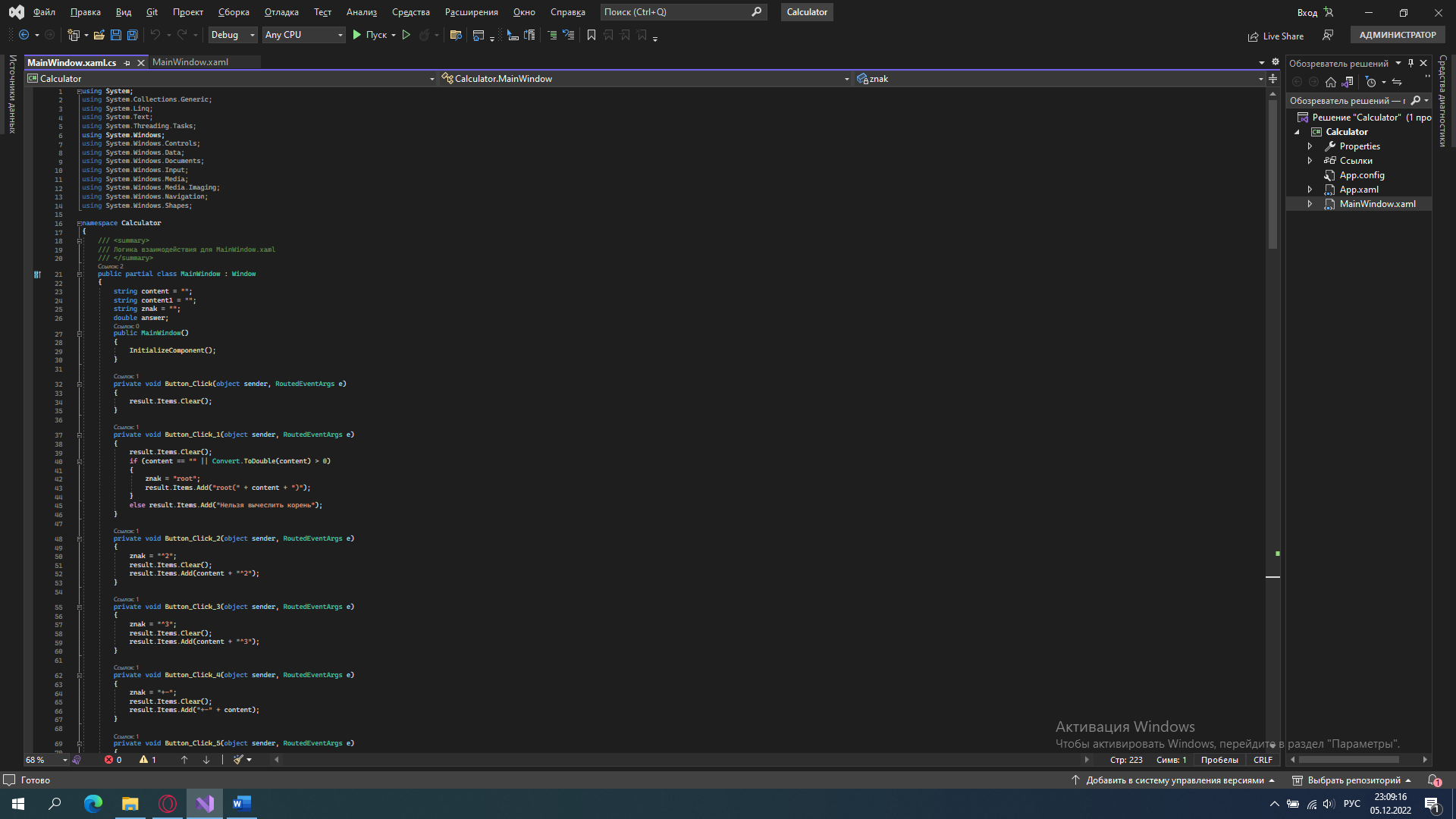
end data

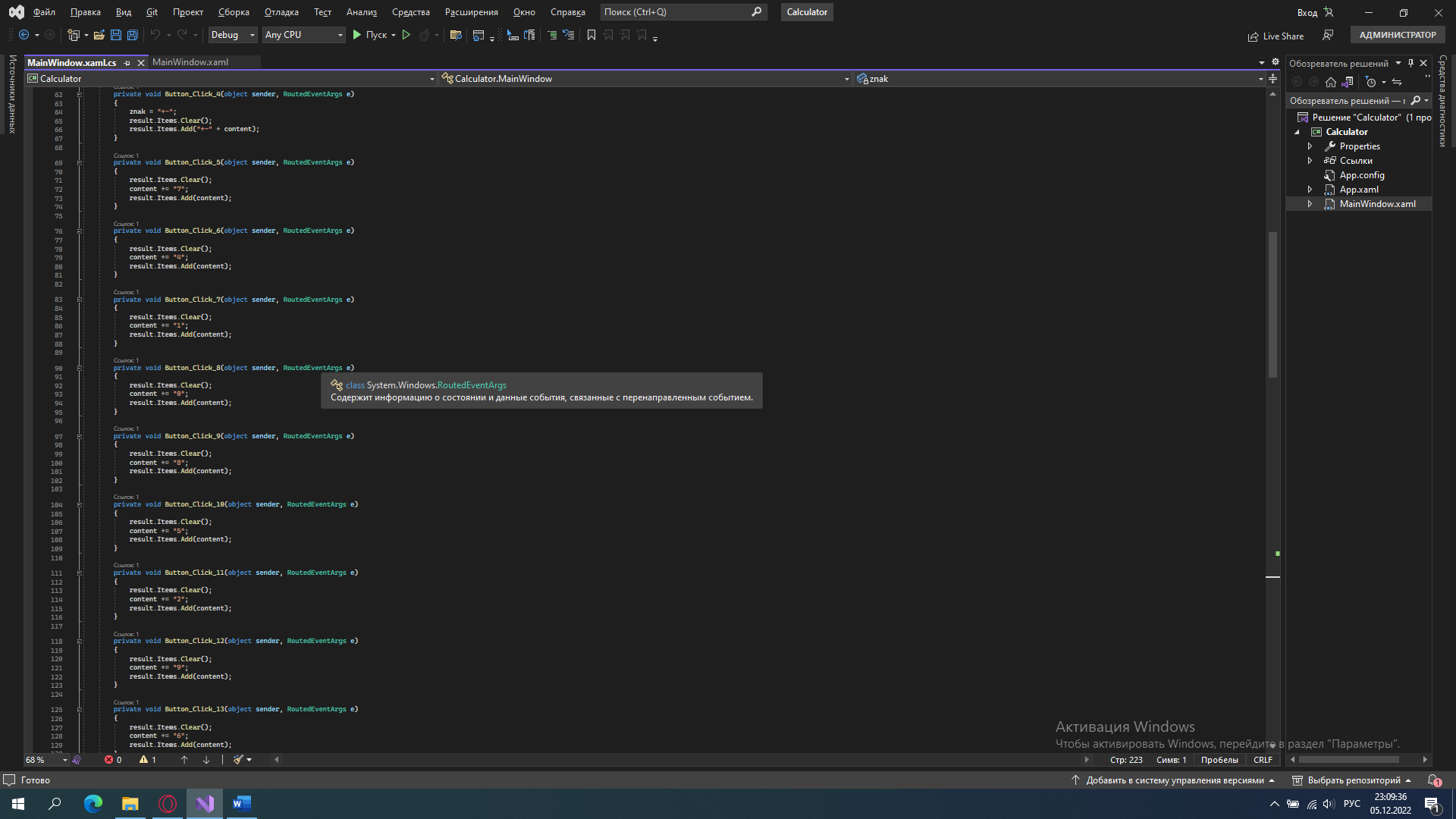
Пример выполнения кода:

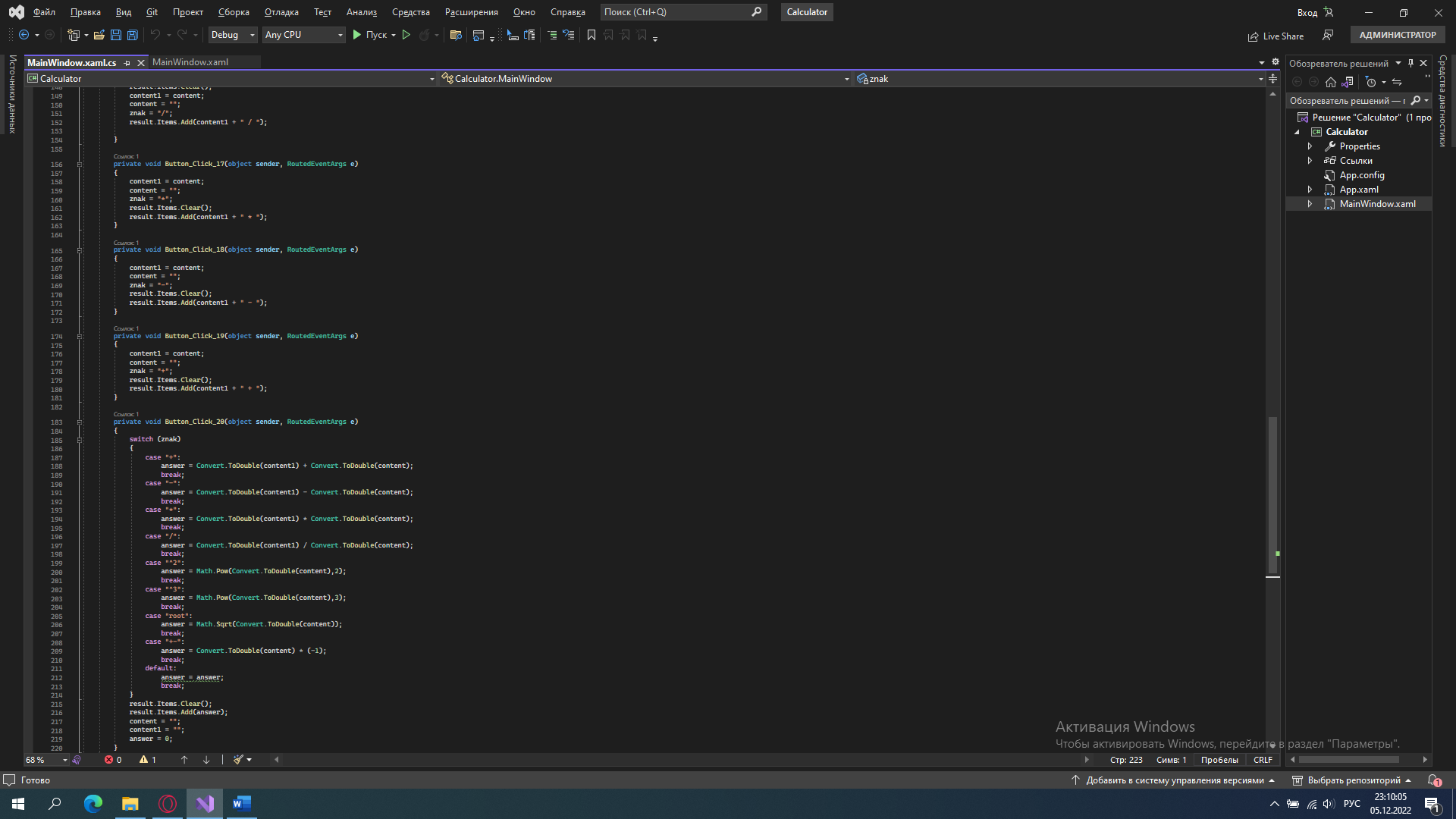
Изображение выглядит как текст

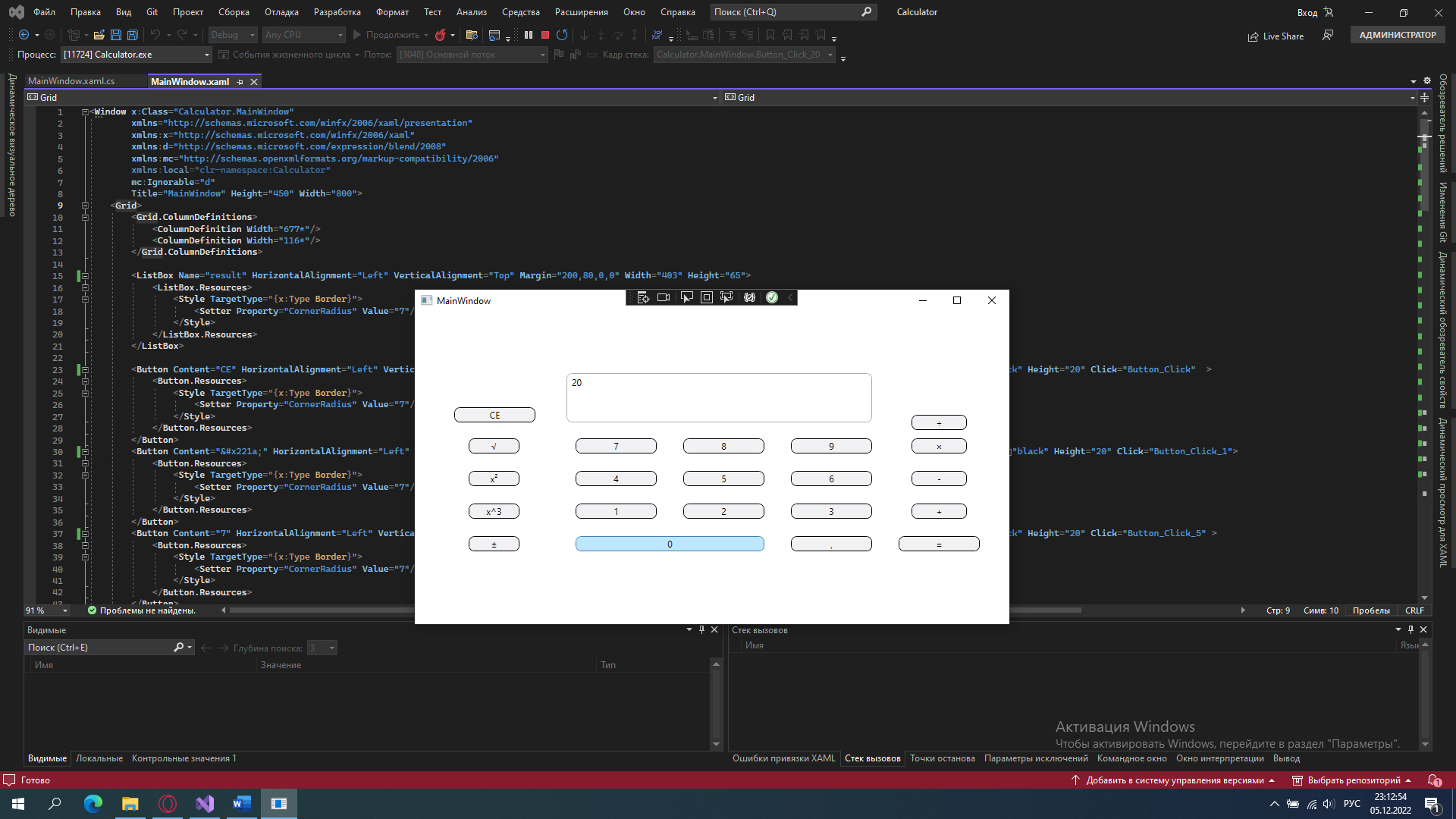
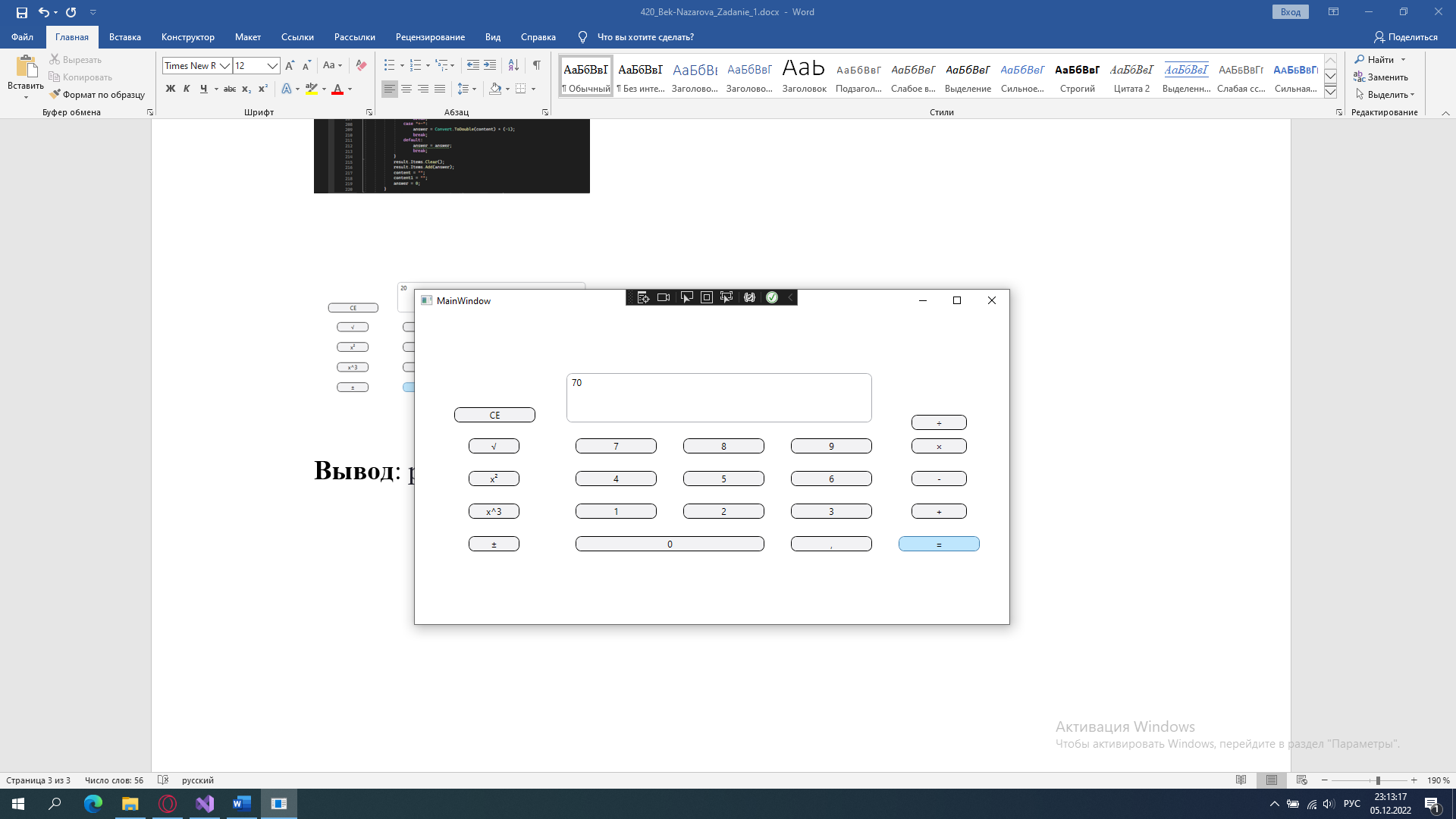
Автоматически созданное описание

**Цель работы**: разработать калькулятор на языке С#.



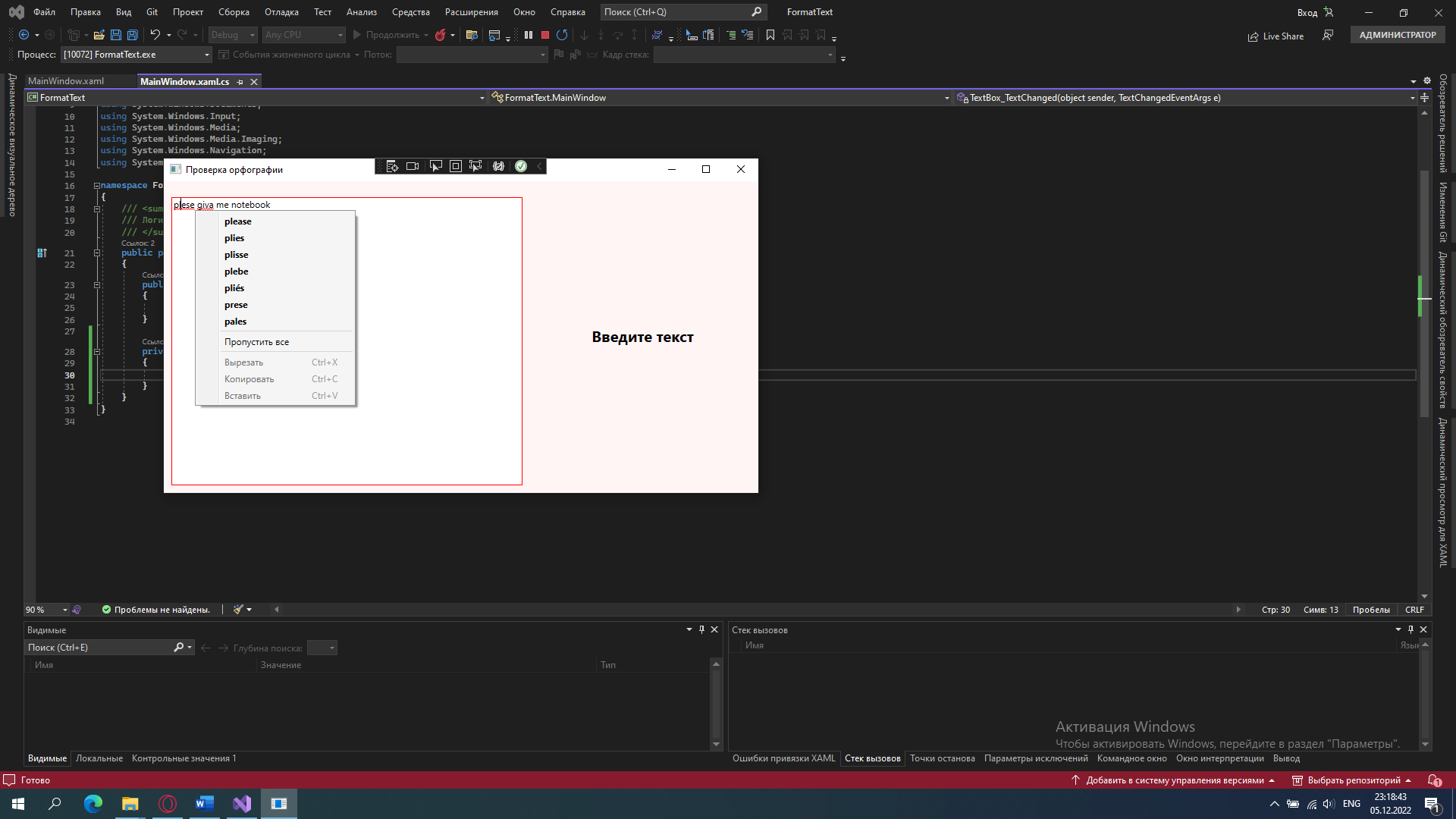






**Вывод**: разработан калькулятор на языке С#.

**Цель работы**: разработать программу на языке С#, которая предлагает пользователю ввести какие-либо слова, предложения в текстовое поле и проверяет их на орфографию.



<Window x:Class="FormatText.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:FormatText"

mc:Ignorable="d"

Title="Проверка орфографии" Height="450" Width="800" Background="#fff5f5">

<Grid>

<TextBox SpellCheck.IsEnabled="True" HorizontalAlignment="Left" Height="380" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="463" Margin="10,21,0,0" BorderBrush="red" TextChanged="TextBox\_TextChanged"/>

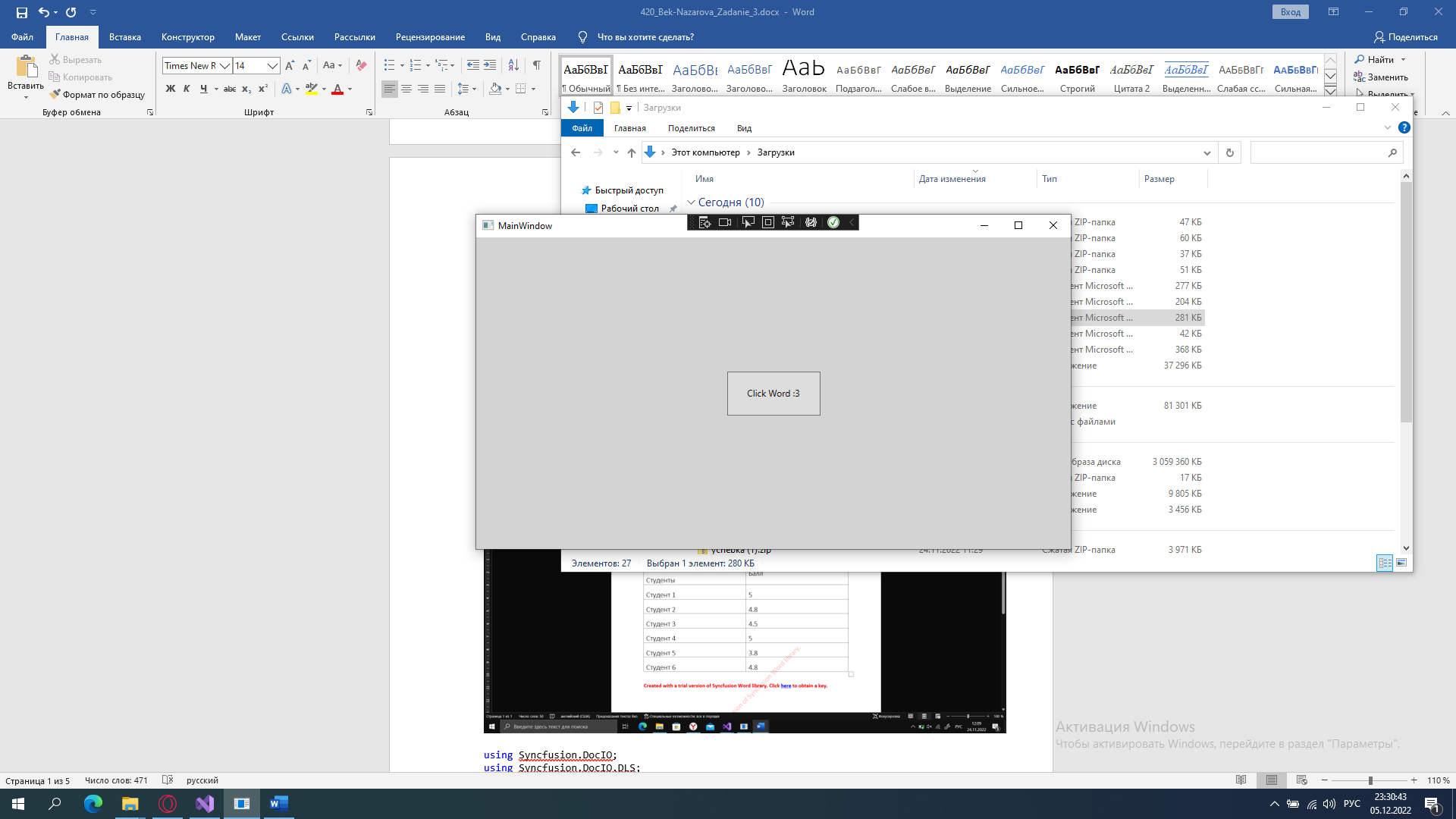
<TextBlock HorizontalAlignment="Left" TextWrapping="Wrap" Text="Введите текст" VerticalAlignment="Top" Margin="400,190,0,0" Width="463" FontSize="20" FontWeight="Bold" TextAlignment="Center" Height="27" Foreground="#000000"/>

</Grid>

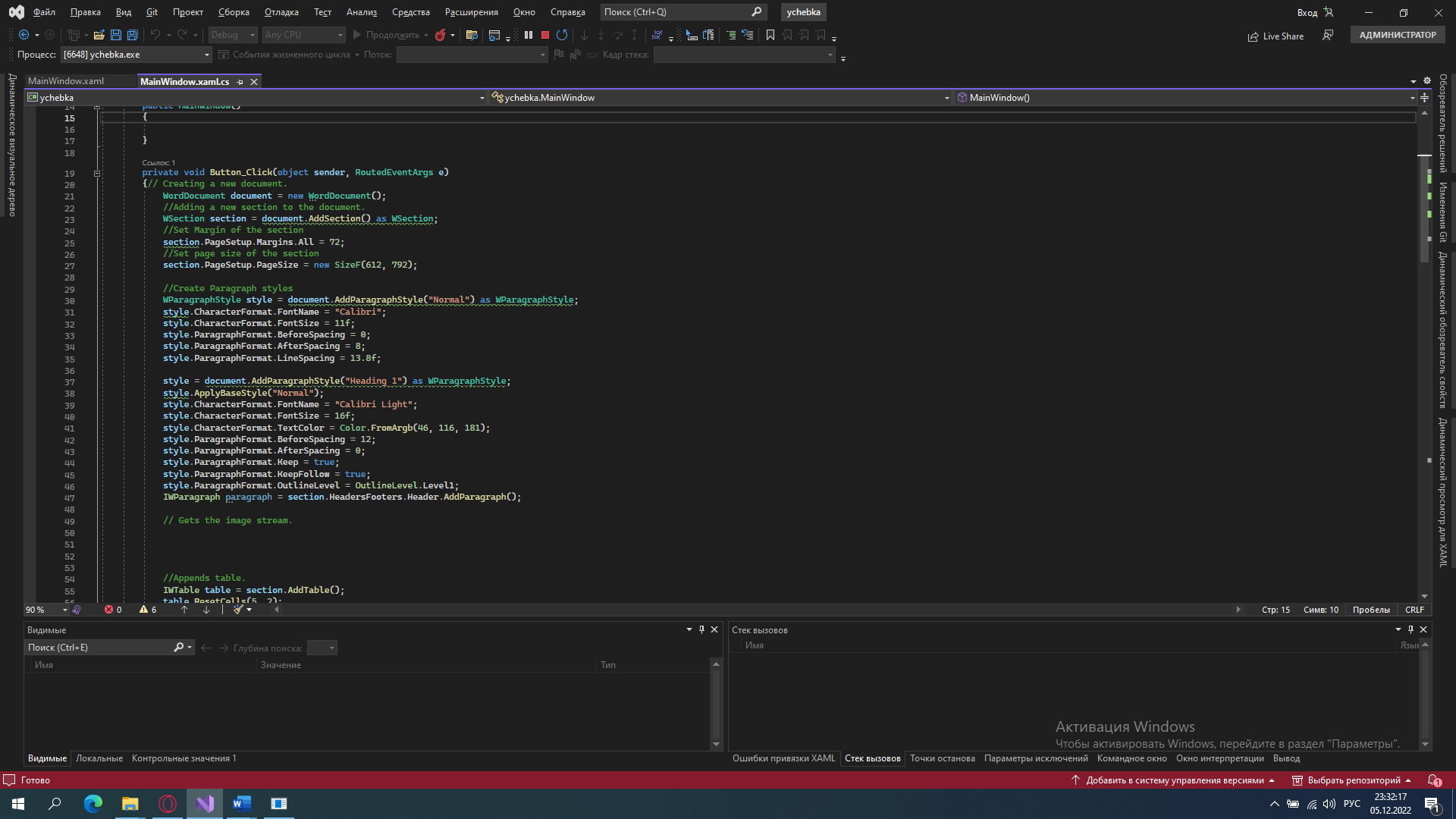
</Window>

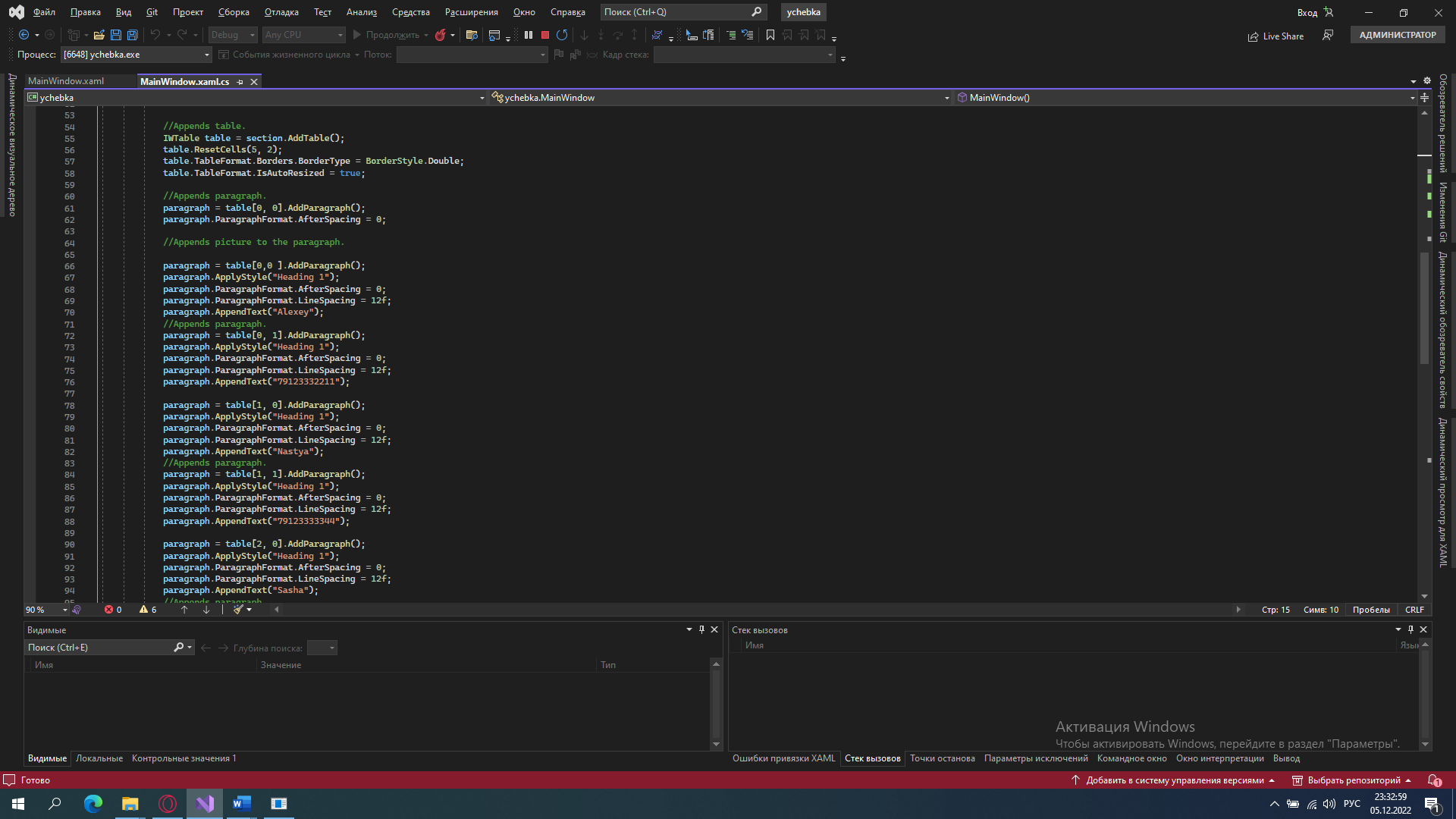
**Вывод**: разработана программа на языке С#, которая предлагает пользователю ввести какие-либо слова, предложения в текстовое поле и проверяет их на орфографию.

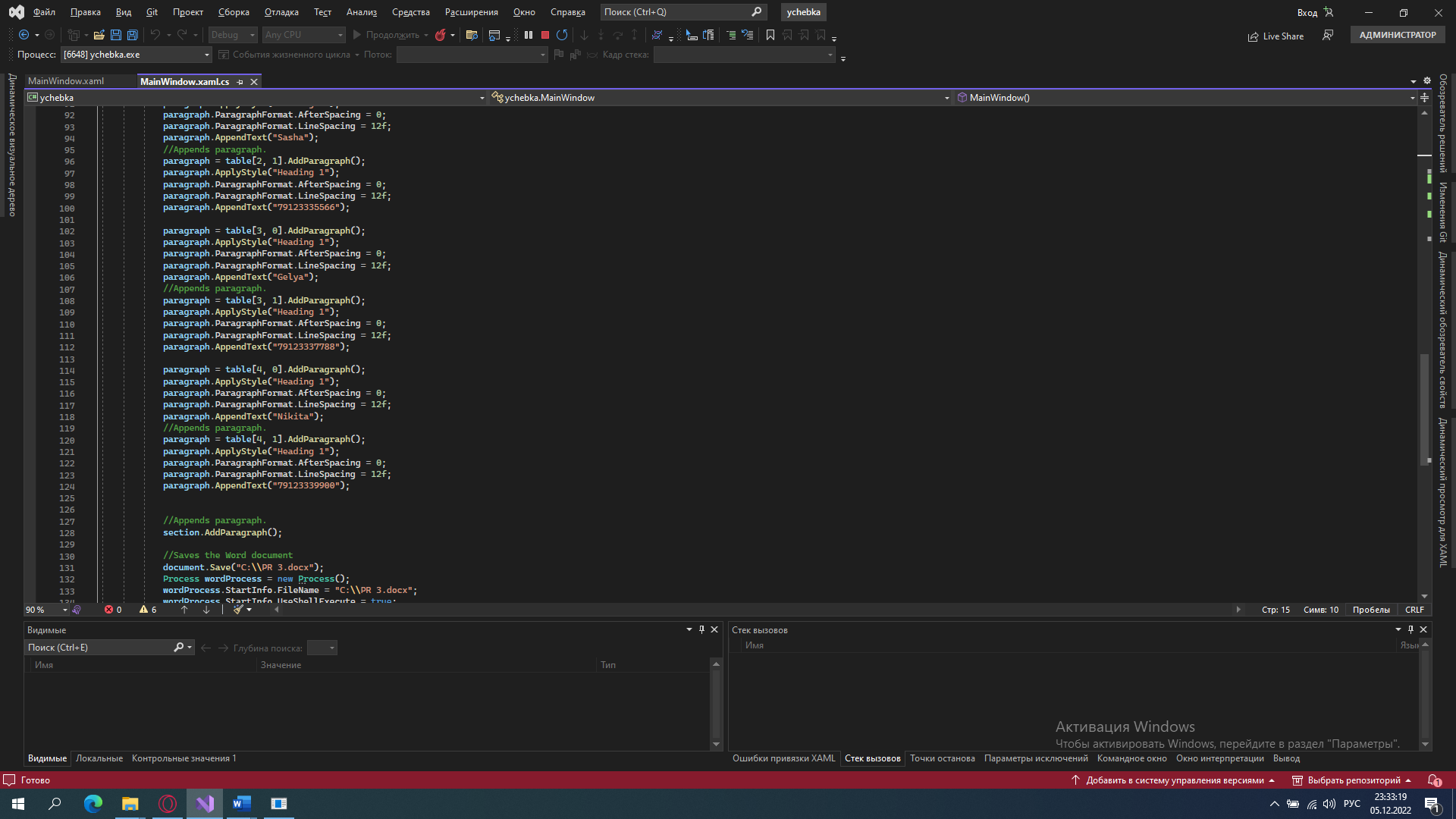
**Цель работы**: необходимо разработать программу, которая при нажатии на кнопку создает таблицу в MS Word.





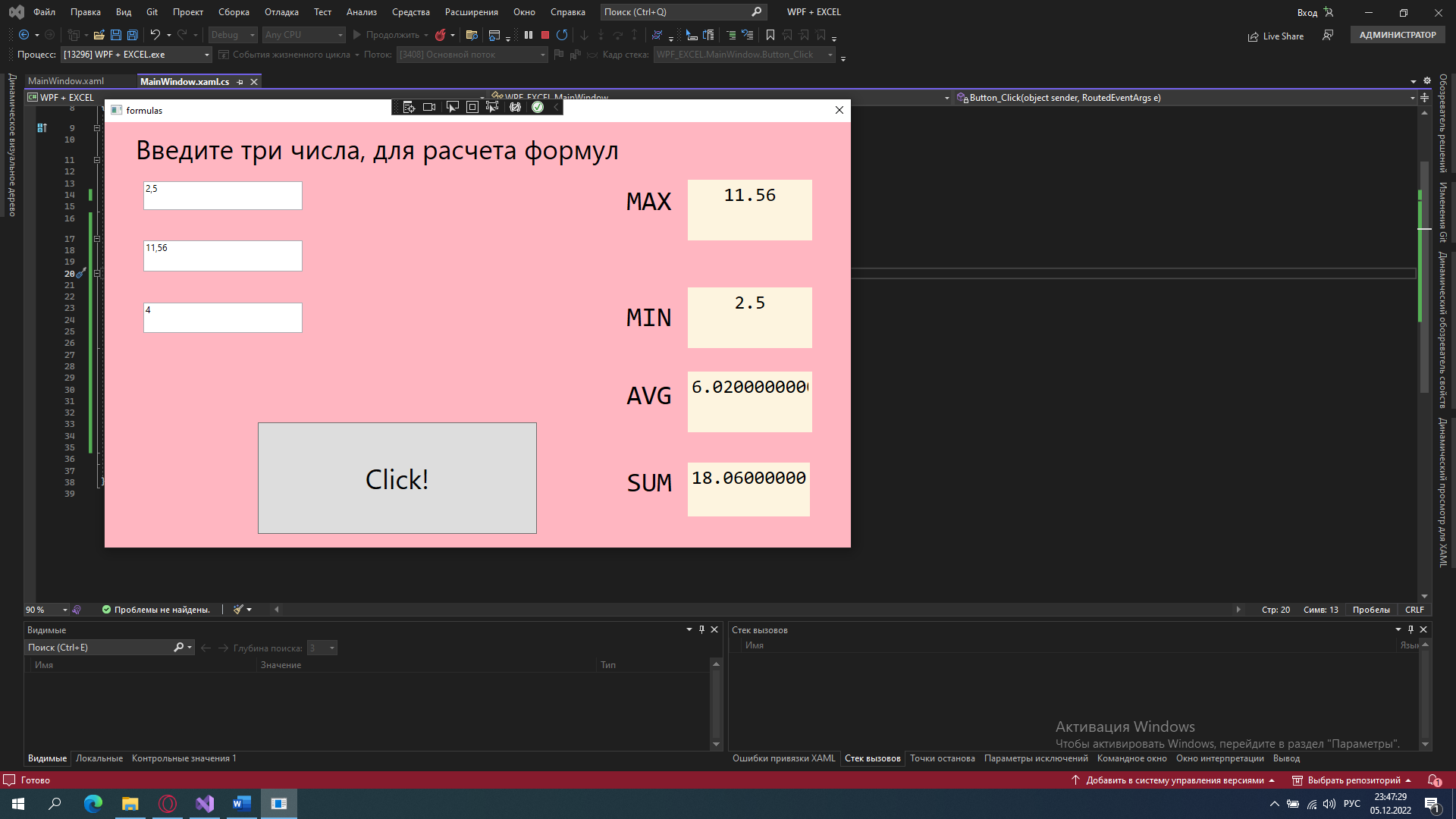


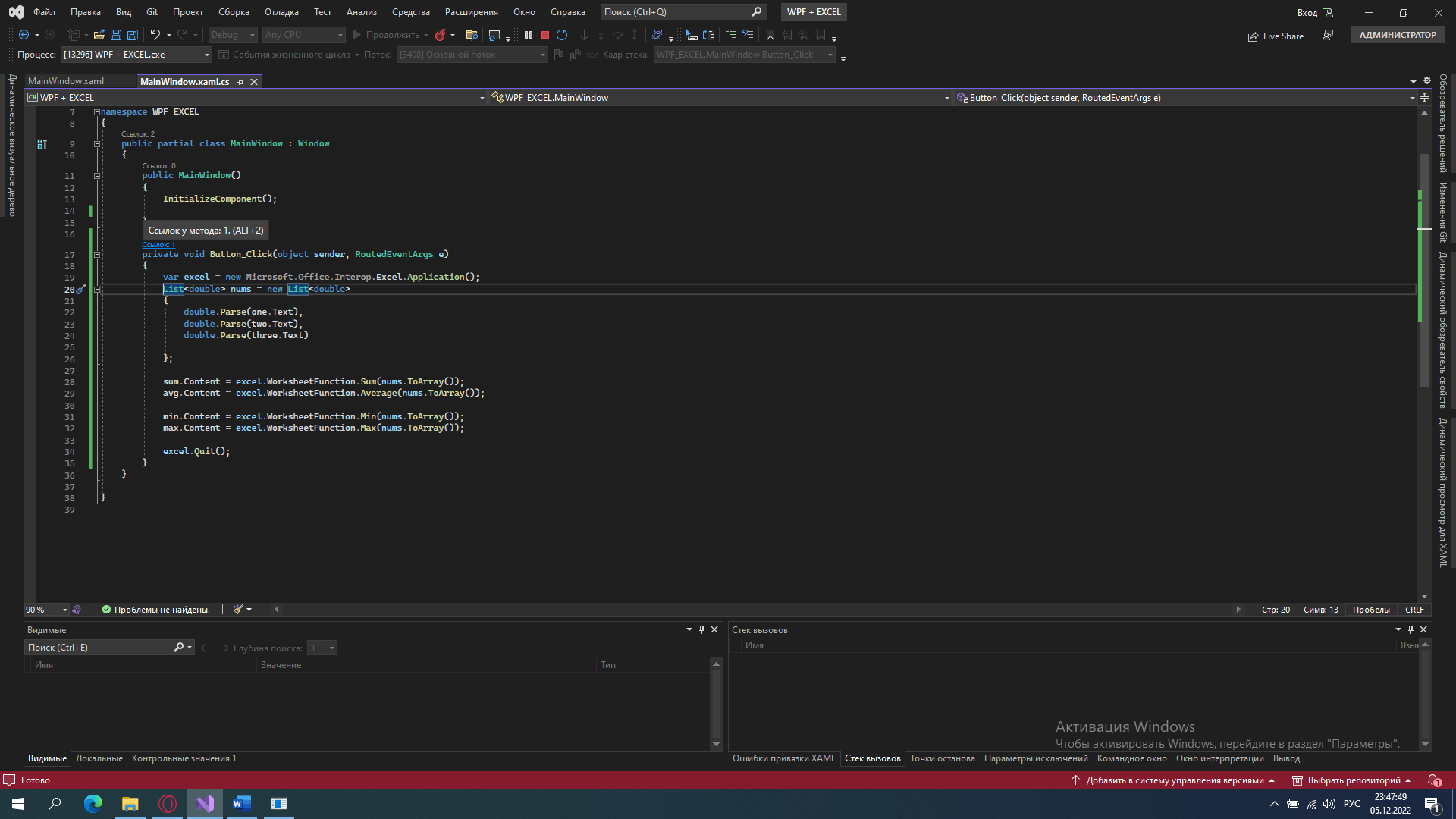




**Вывод**: разработана программа, которая при нажатии на кнопку создает таблицу в MS Word.

**Цель работы**: необходимо разработать программу с использованием функций MS Excel не используя функцию нахождения числа Pi.

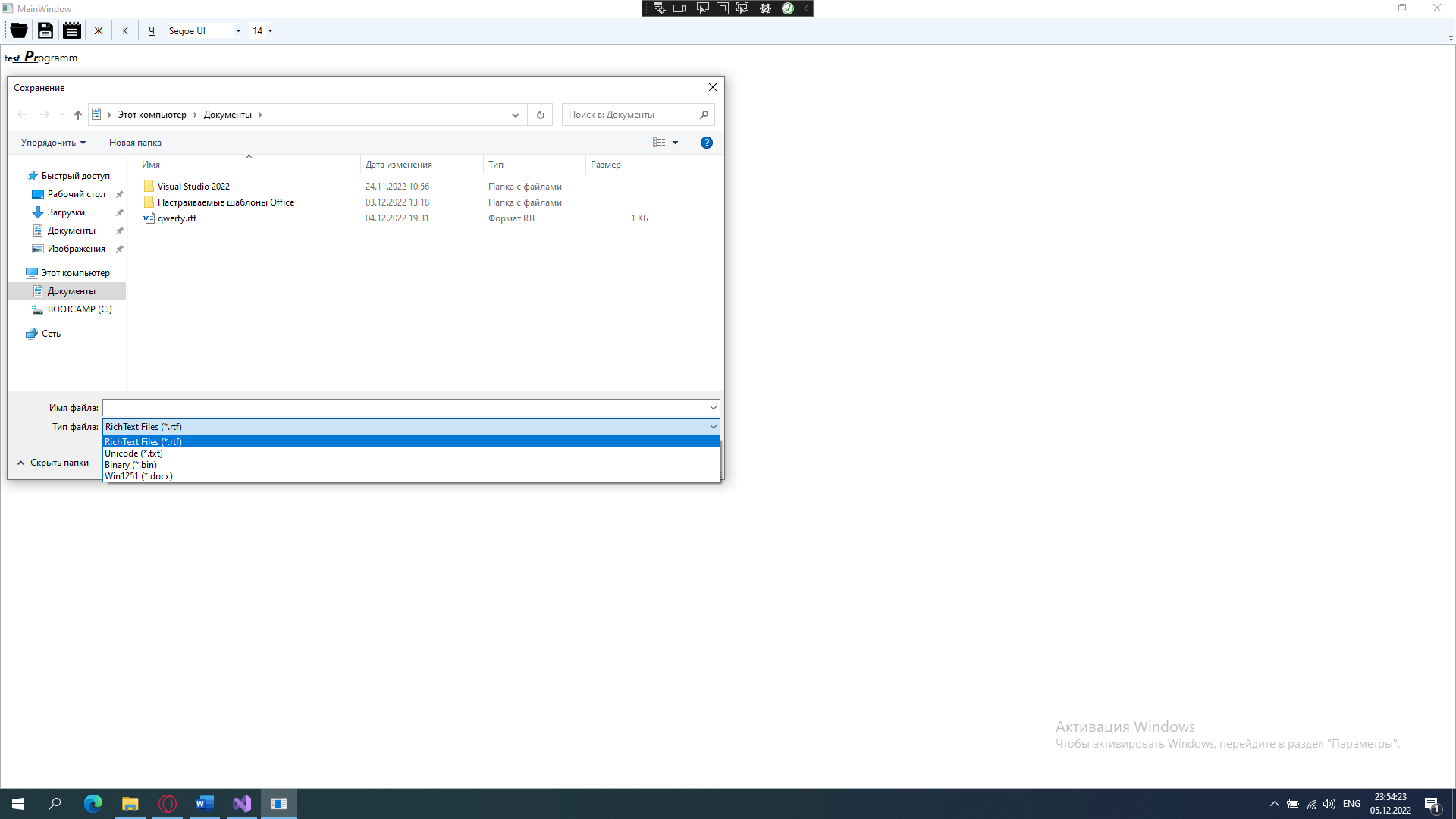


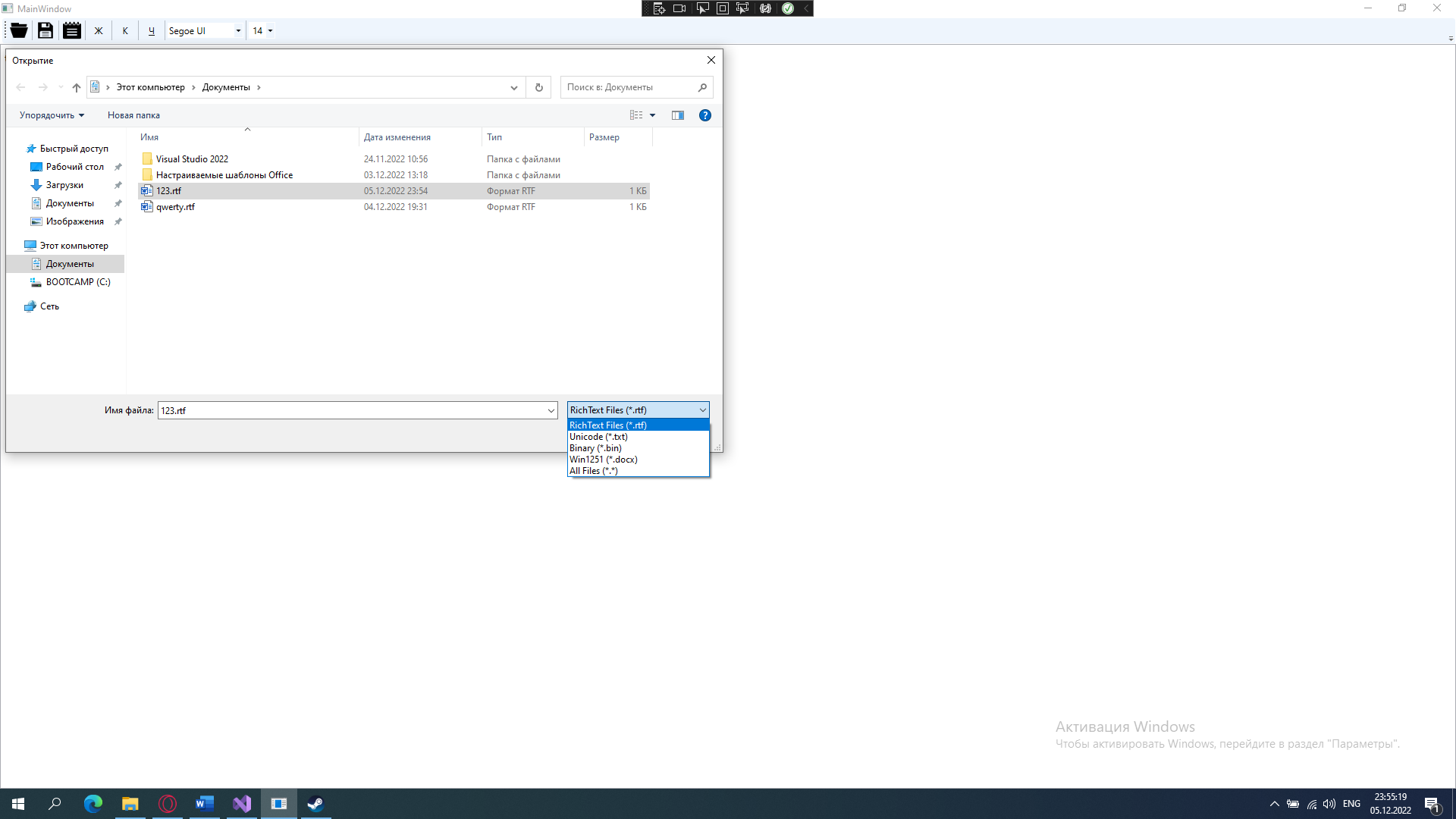


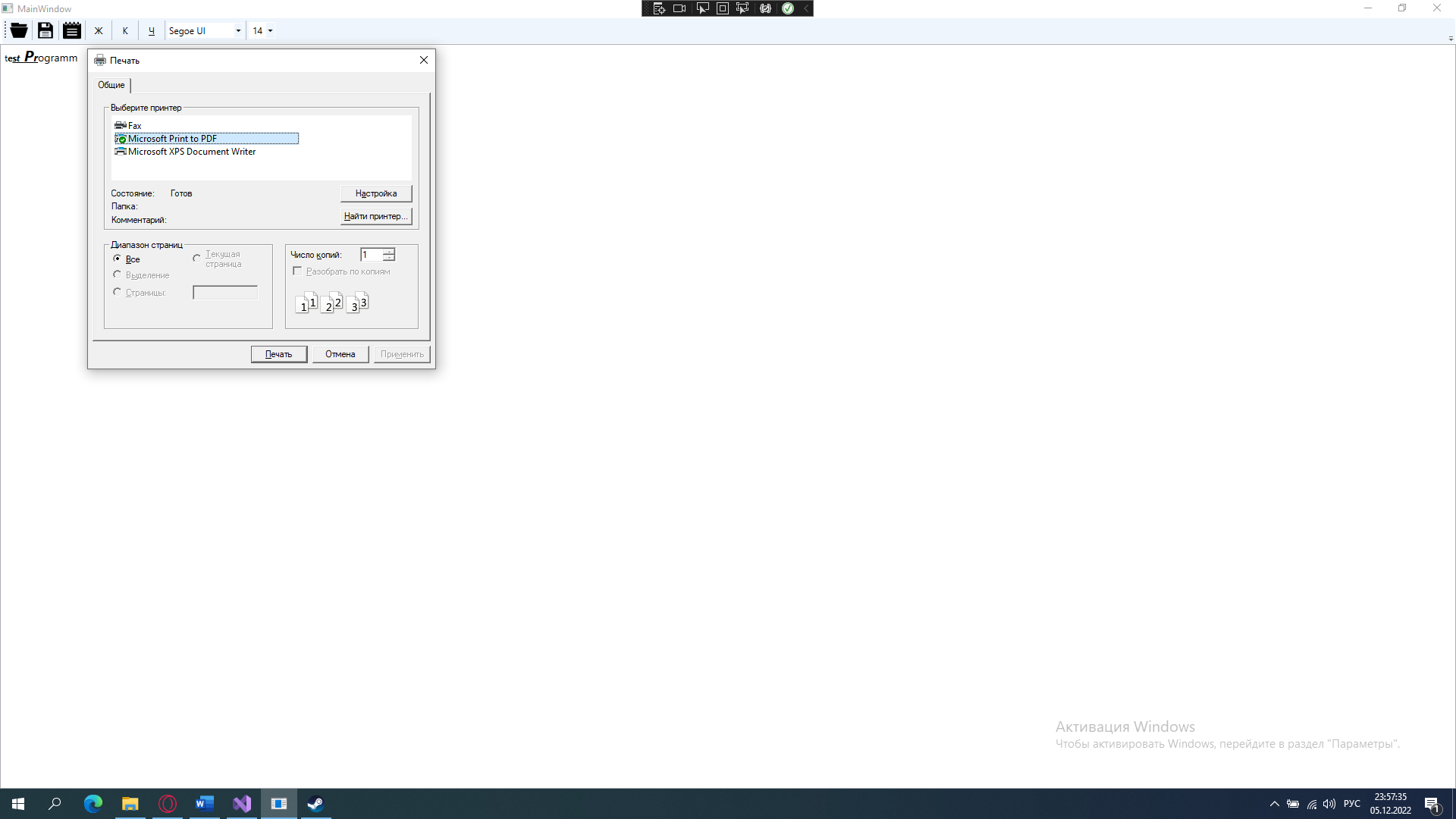
**Вывод**: разработана программа с использованием функций MS Excel не используя функцию нахождения числа Pi.

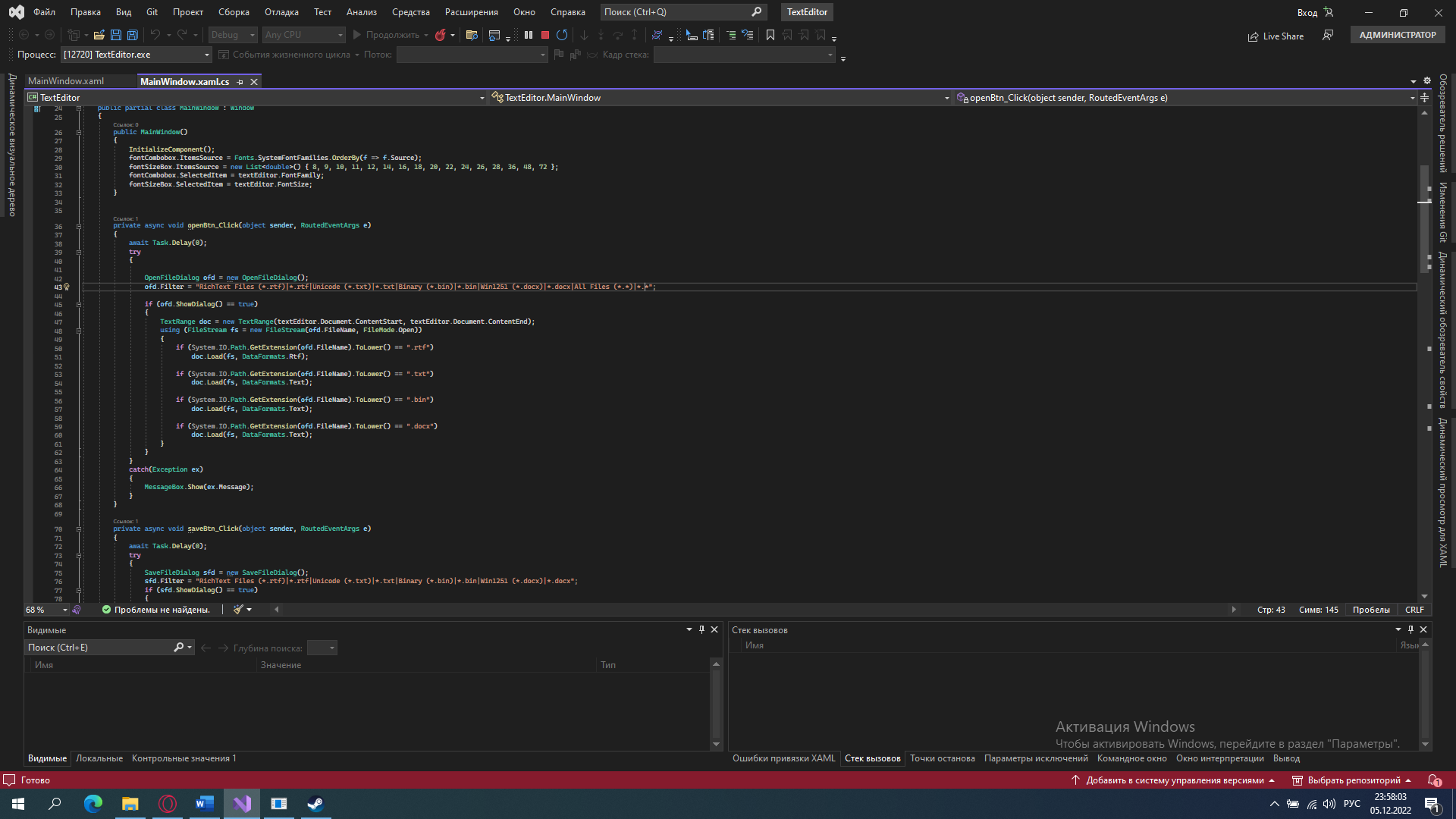
**Цель работы**: написать программу на языке C#, которая реализует работу с файлами в различных кодировках.

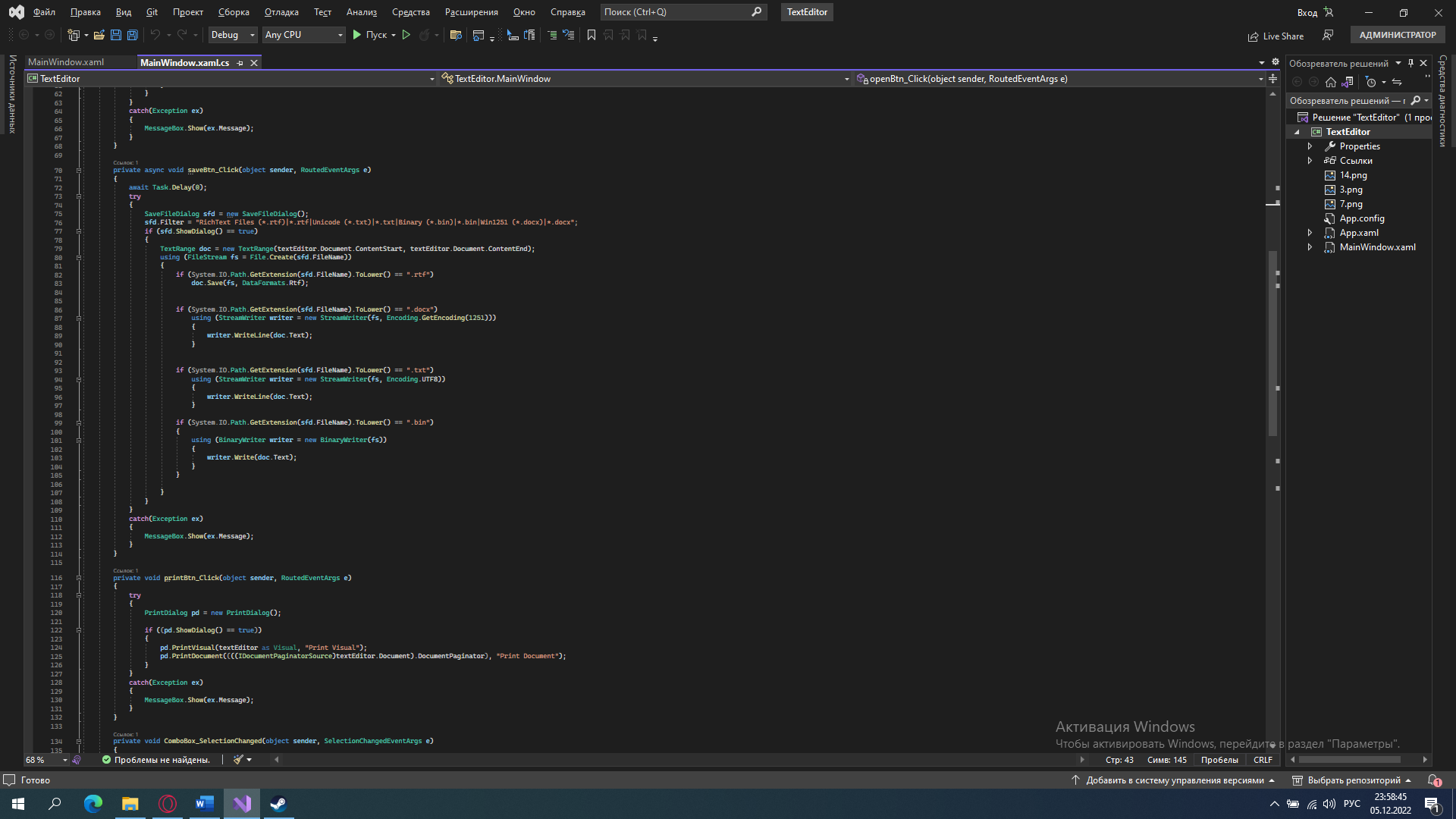


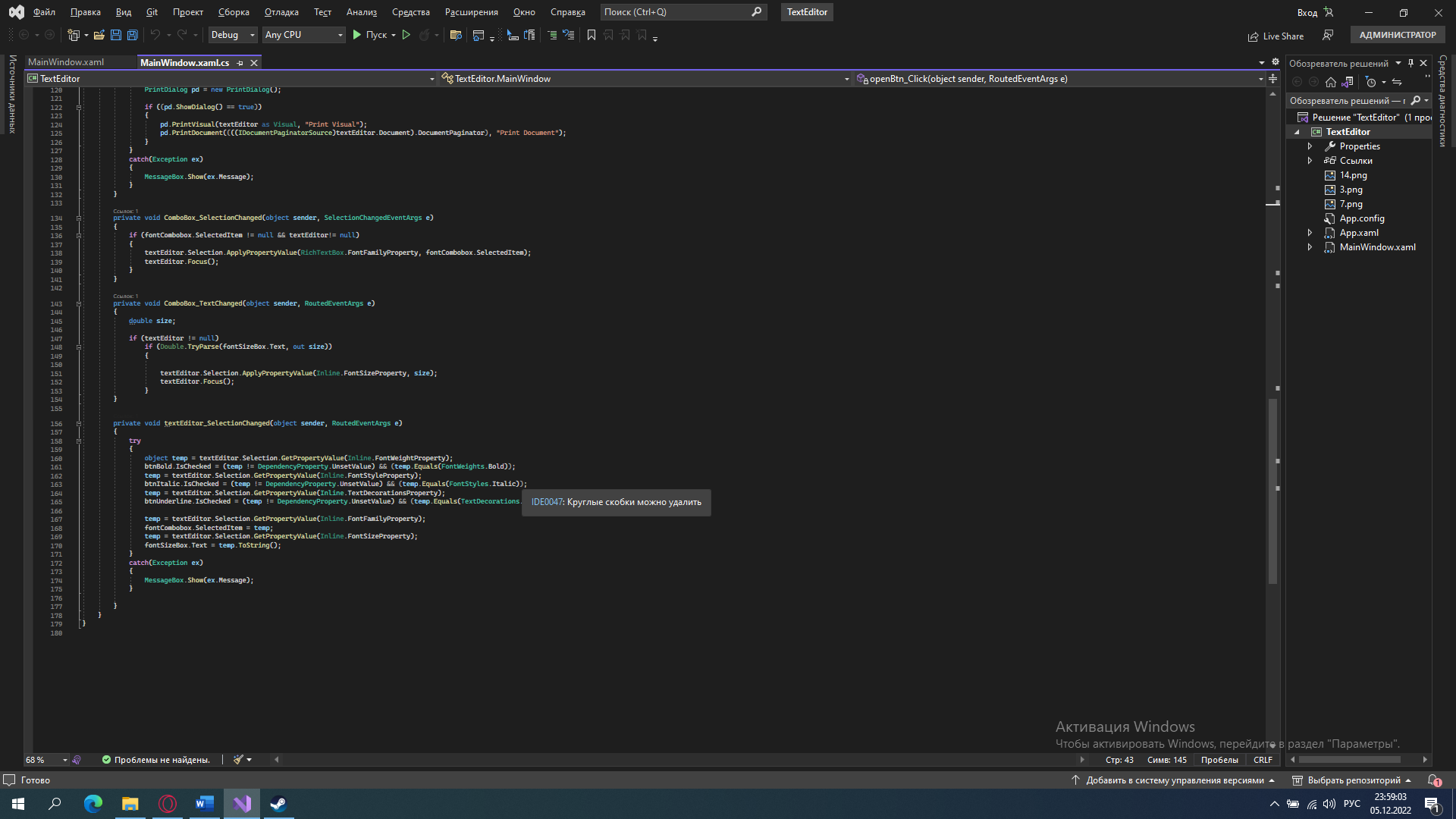












**Вывод**: написана программа на языке C#, которая реализует работу с файлами в различных кодировках.