Министерство образования Республики Беларусь

Учреждение образования

"Брестский государственный технический университет"

Кафедра ИИТ

Лабораторная работа №9

За 2ой семестр

По дисциплине: "ОАиП"

Тема: «Пользовательские функции»

Выполнил:

Студент 1-го курса АС-59

(подгруппа 2)

Левоцкий Н.

Цыганов Е.

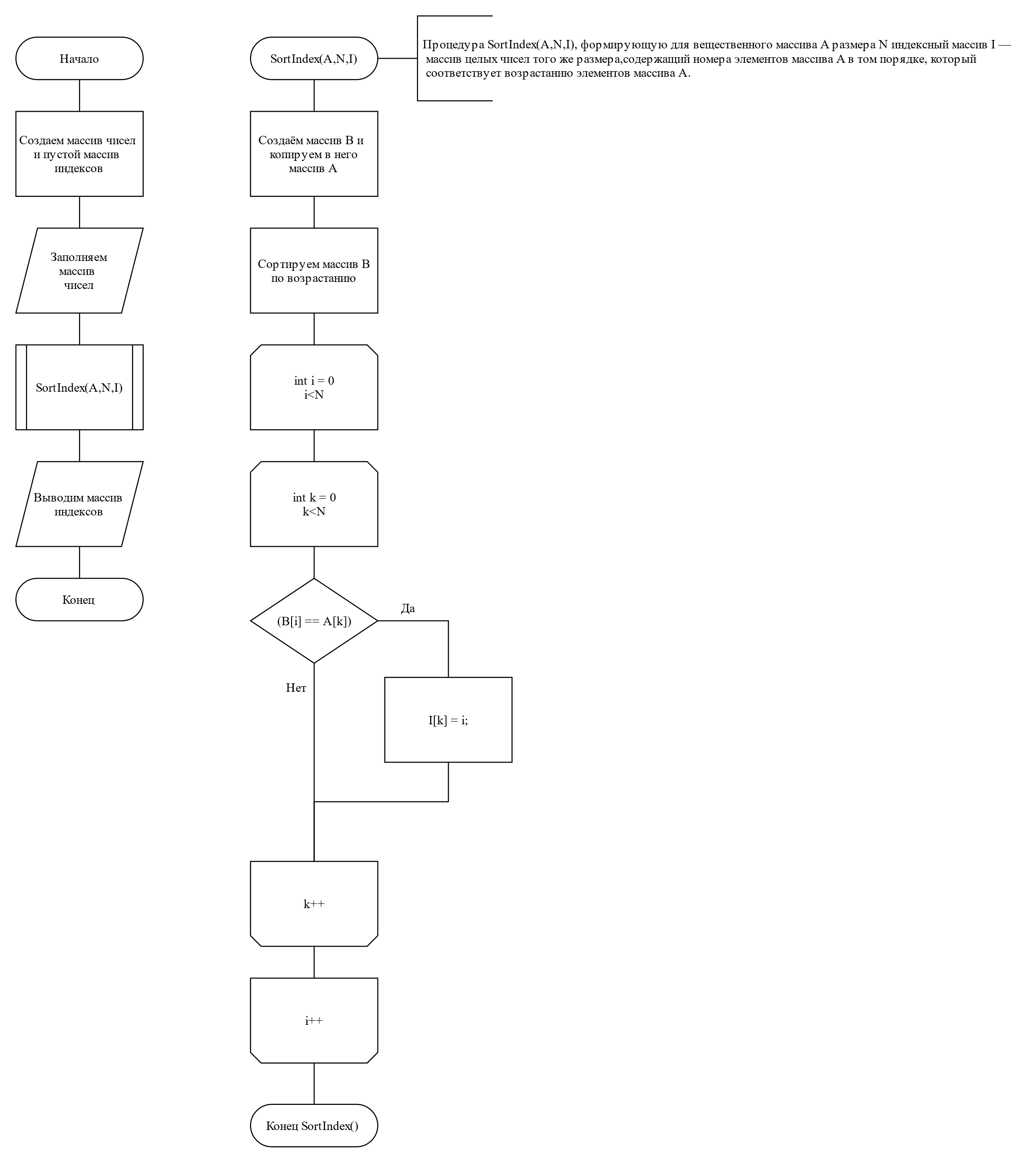
Проверила:

Гирель Т. Н.

Брест 2021

**Цель работы:** Изучить основные принципы написания пользовательских функций, ознакомиться с возможностями передачи данных в функции и получения результата по итогам работы функции. Реализовать собственные функции для обработки данных составных и простых типов.

**Задание 1.**



12. Описать процедуру SortIndex(A, N, I), формирующую для вещественного массива A размера N индексный массив I — массив целых чисел того же размера,

содержащий номера элементов массива A в том порядке, который соответствует возрастанию элементов массива A (сам массив A при этом не изменяется).

Индексный массив I является выходным параметром. С помощью этой процедуры создать индексные массивы для массивов A,B, C размера NA, NB, NC соответственно.

#include<string.h>

#include<iostream>

#pragma warning(disable : 4996)

using namespace std;

int size;

cout << "Number of elements: "; cin >> size;

float\* arr = new float[size];

cout << "Elements: \n";

for (int i = 0; i < size; i++)

{

cout << "-> ";

cin >> arr[i];

}

int \*index\_arr = new int[size];

cout << "Array: ";

for (int i = 0; i < size; i++)

{

cout << arr[i] << " ";

}

cout << endl;

sortindex(arr, size, index\_arr);

cout << endl;

cout << endl << " Index array\n";

for (int i = 0; i < size; i++)

{

cout << index\_arr[i] << " ";

}

void sortindex(float A[], int N, int I[])

{

float\* B = new float[N];

for (int i = 0; i < N; i++)

{

B[i] = A[i];

}

for (int i = 0; i < N; i++)

{

for ( int k = i; k < N; k++)

{

if (B[k] < B[i])

{

float temp = B[i];

B[i] = B[k];

B[k] = temp;

}

}

}

cout << endl << "Sorted array \n";

for (int i = 0; i < N; i++)

{

cout << B[i] << " ";

}

for (int i = 0; i < N; i++)

{

for (int k = 0; k < N; k++)

{

if (B[i] == A[k])

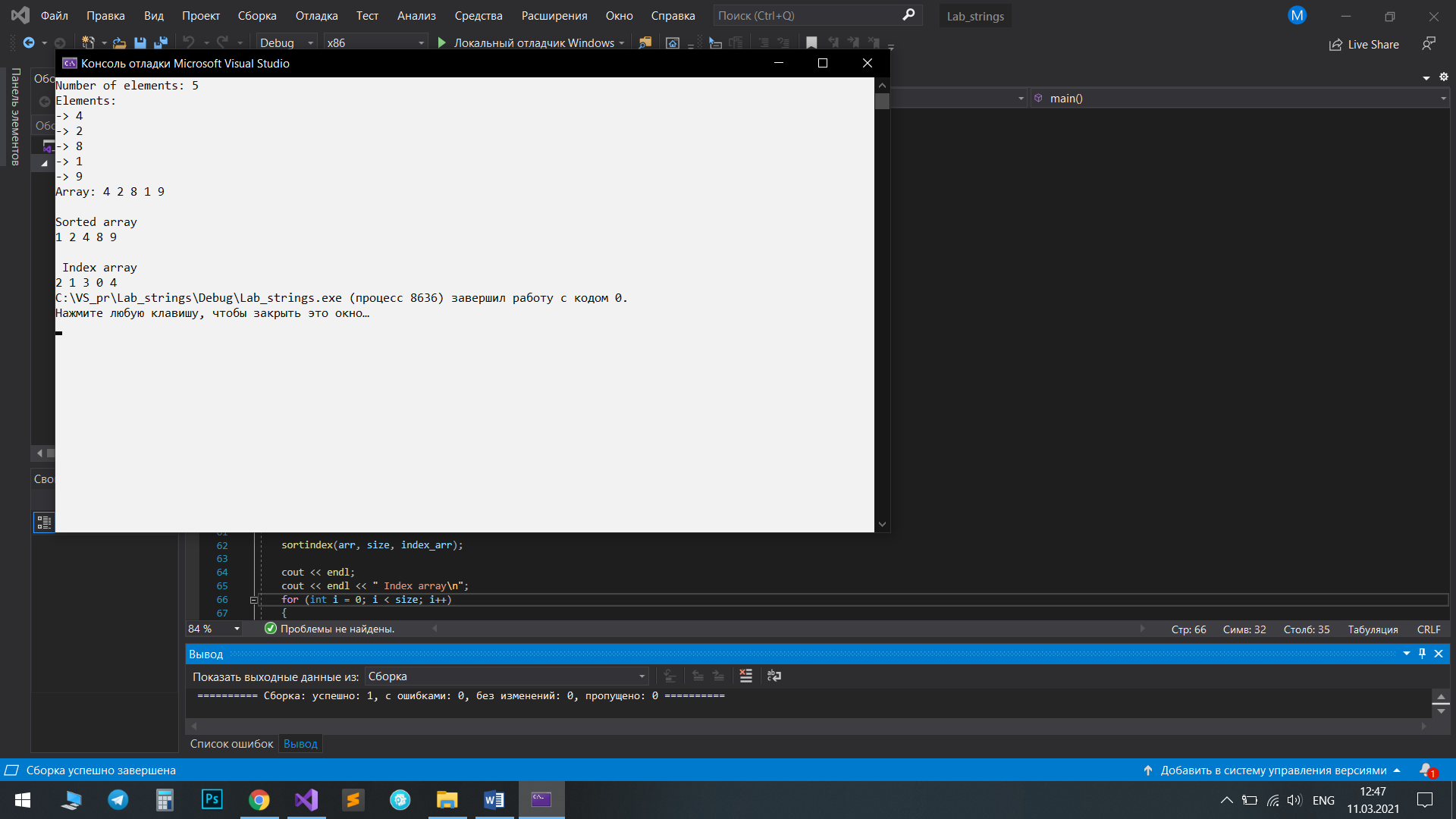
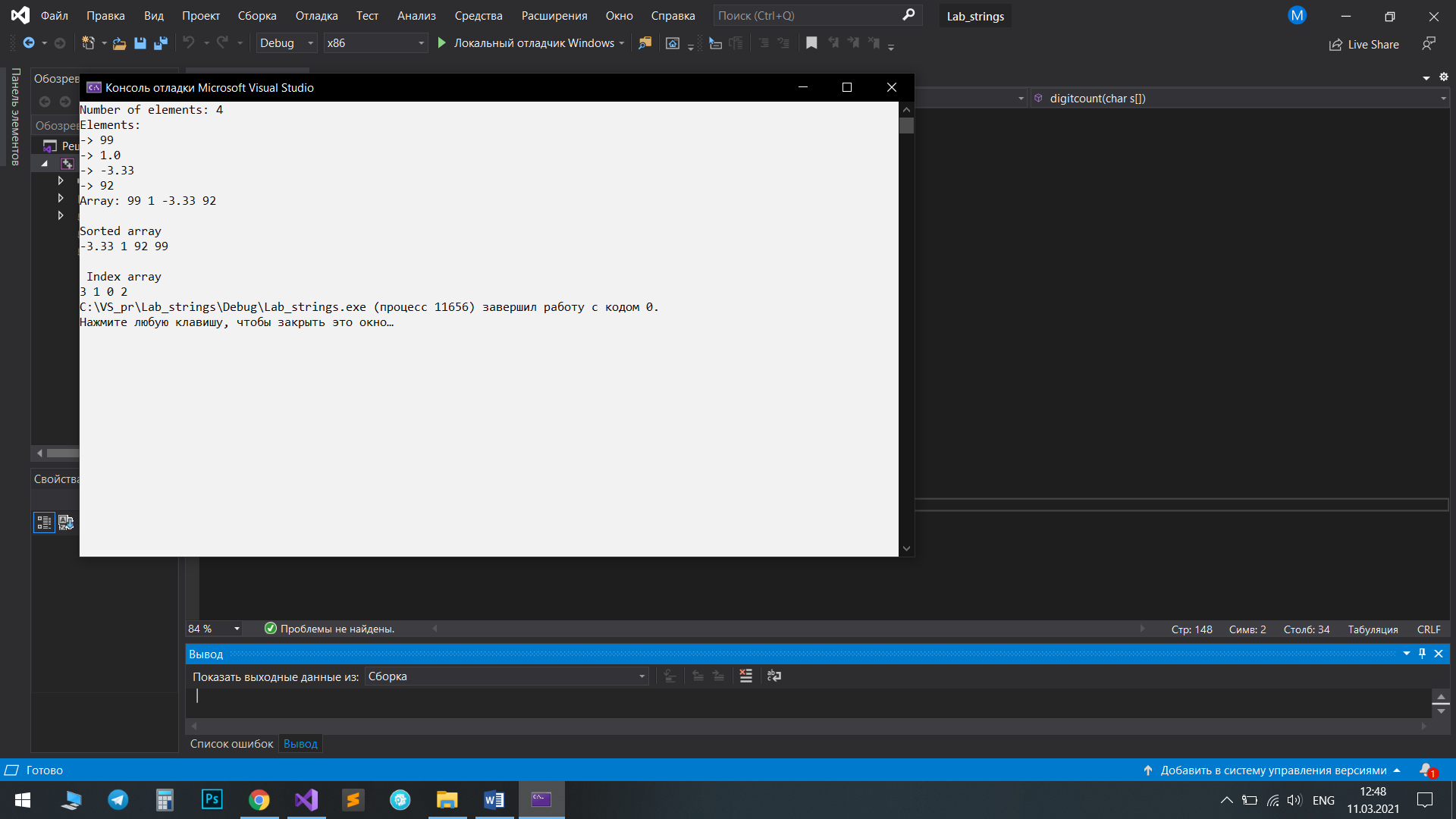
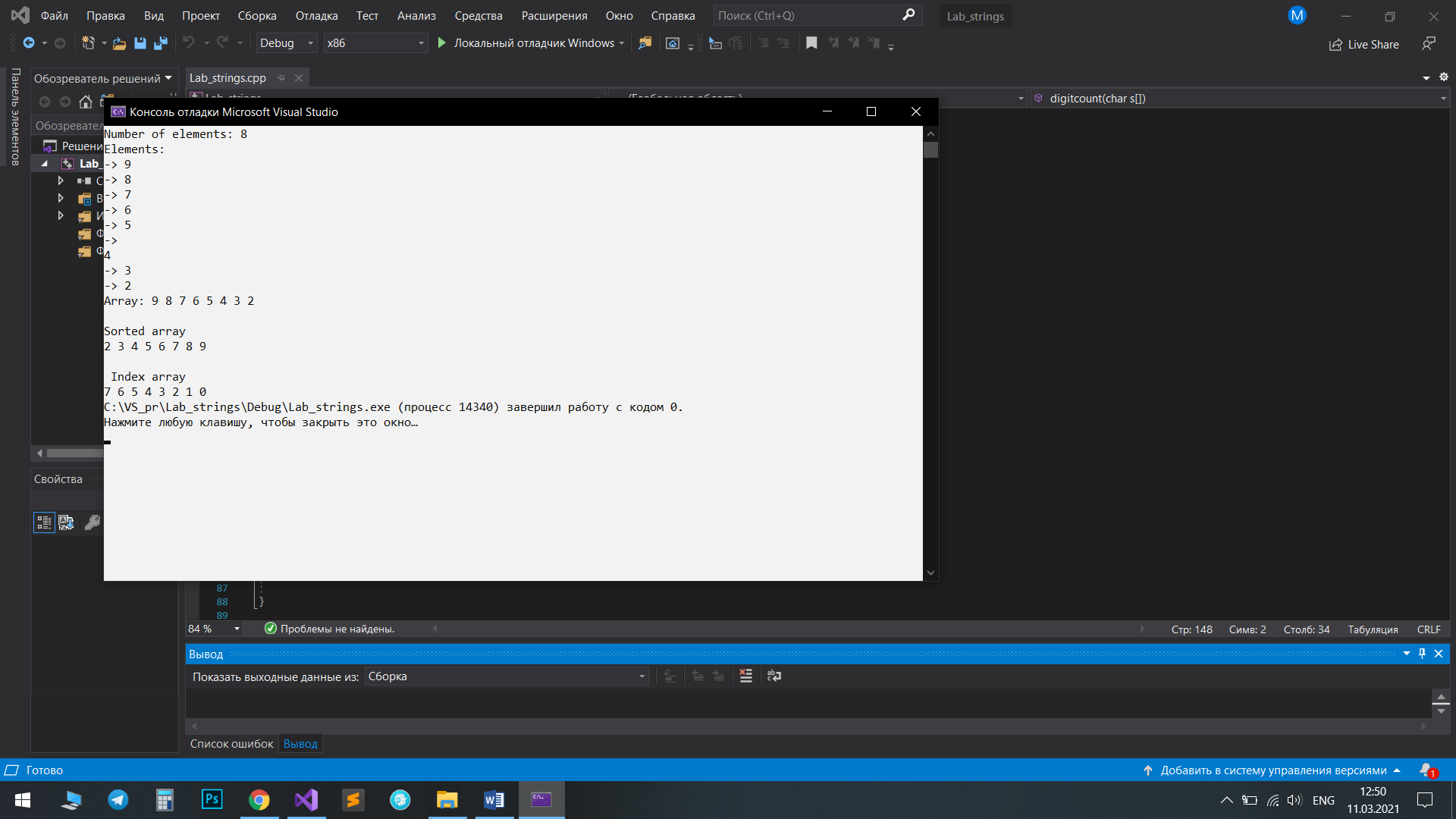
I[k] = i;

}

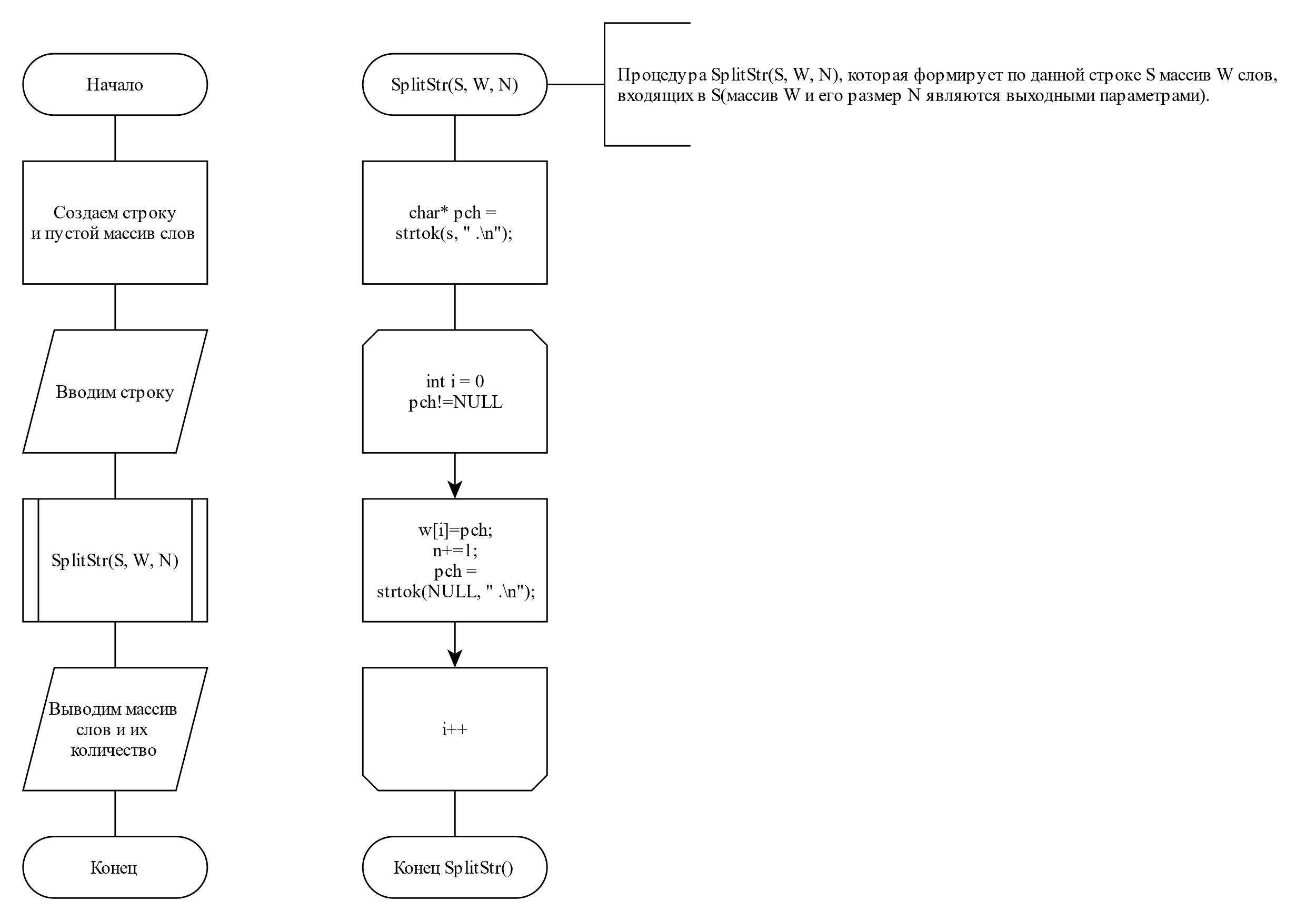
}

delete[] B;

}

**Задание 2.**



12. Описать процедуру SplitStr(S, W, N), которая формирует по данной строке S массив W слов, входящих в S(массив W и его размер N являются выходными параметрами).

Словом считается набор символов, не содержащий пробелов и ограниченный пробелами или началом / концом строки; предполагается, что строка S содержит не более 10 слов.

Используя функцию SplitStr, найти количество слов N, содержащихся в данной строке S, и сами эти слова.

#include<string.h>

#include<iostream>

#pragma warning(disable : 4996)

using namespace std;

int splitstr(char s[], char\*\* w, int &n);

int main()

{

char str[50]; //вводимая строка

cout << "Input string: ";

gets\_s(str);

int number = 0; //изначальное количество слов, которое мы передадим в функцию и она присвоит действительное количество слов

char\*\* words = new char\*[10]; //пустой массив слов, который мы передадим в функцию и она заполнит его

splitstr(str, words, number);

cout << "\tWords:\t\n"; //выводим слова

for (int i = 0; i < number; i++)

{

cout << words[i] << endl;

}

cout << "Number: " << number << endl; //выводим количество

}

int splitstr(char s[], char\*\* w, int& n)

{

char\* pch = strtok(s, " .\n"); //делим строку

for (int i = 0;pch!=NULL; i++)

{

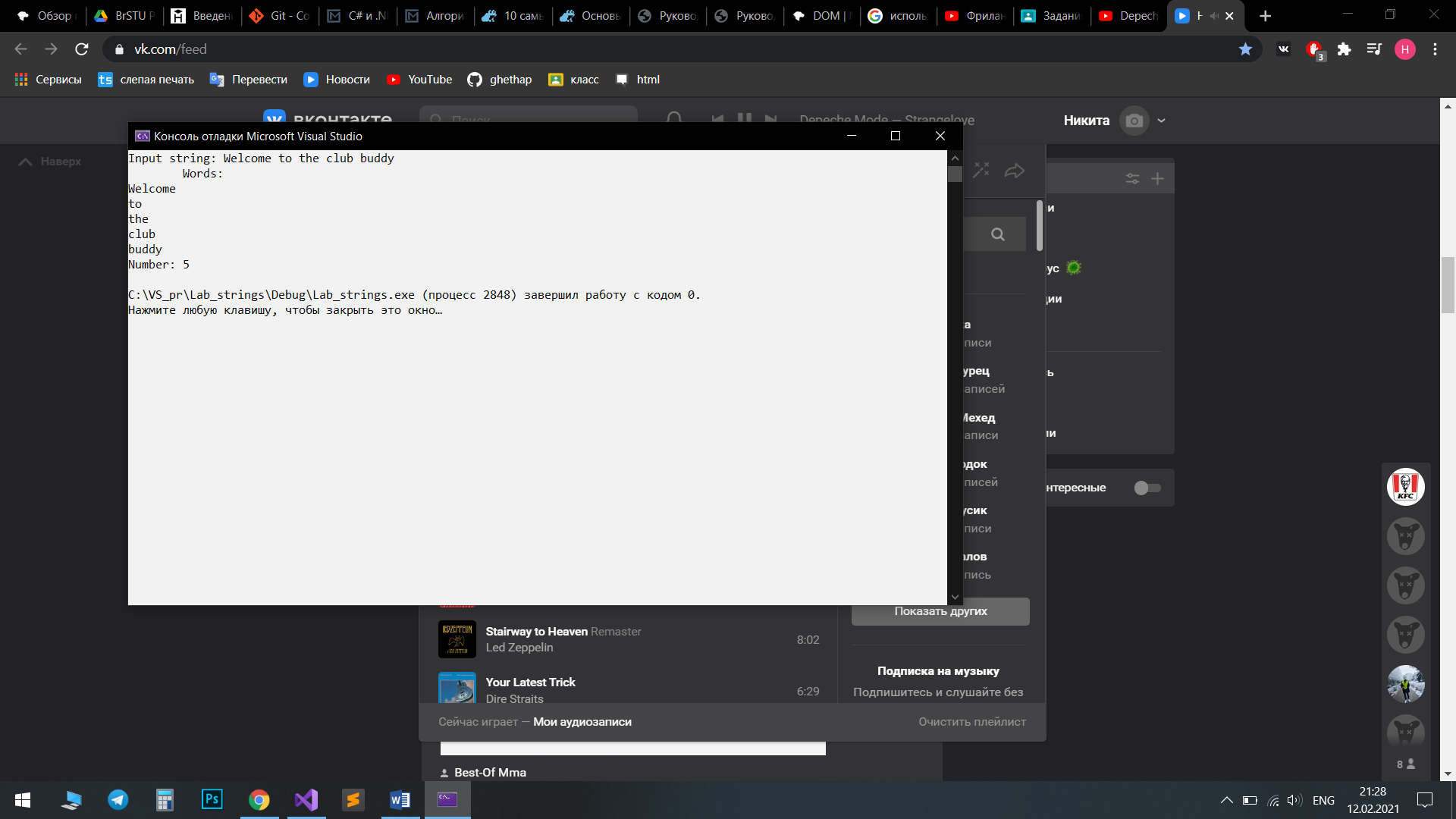
w[i]=pch; //заполняем массив слов

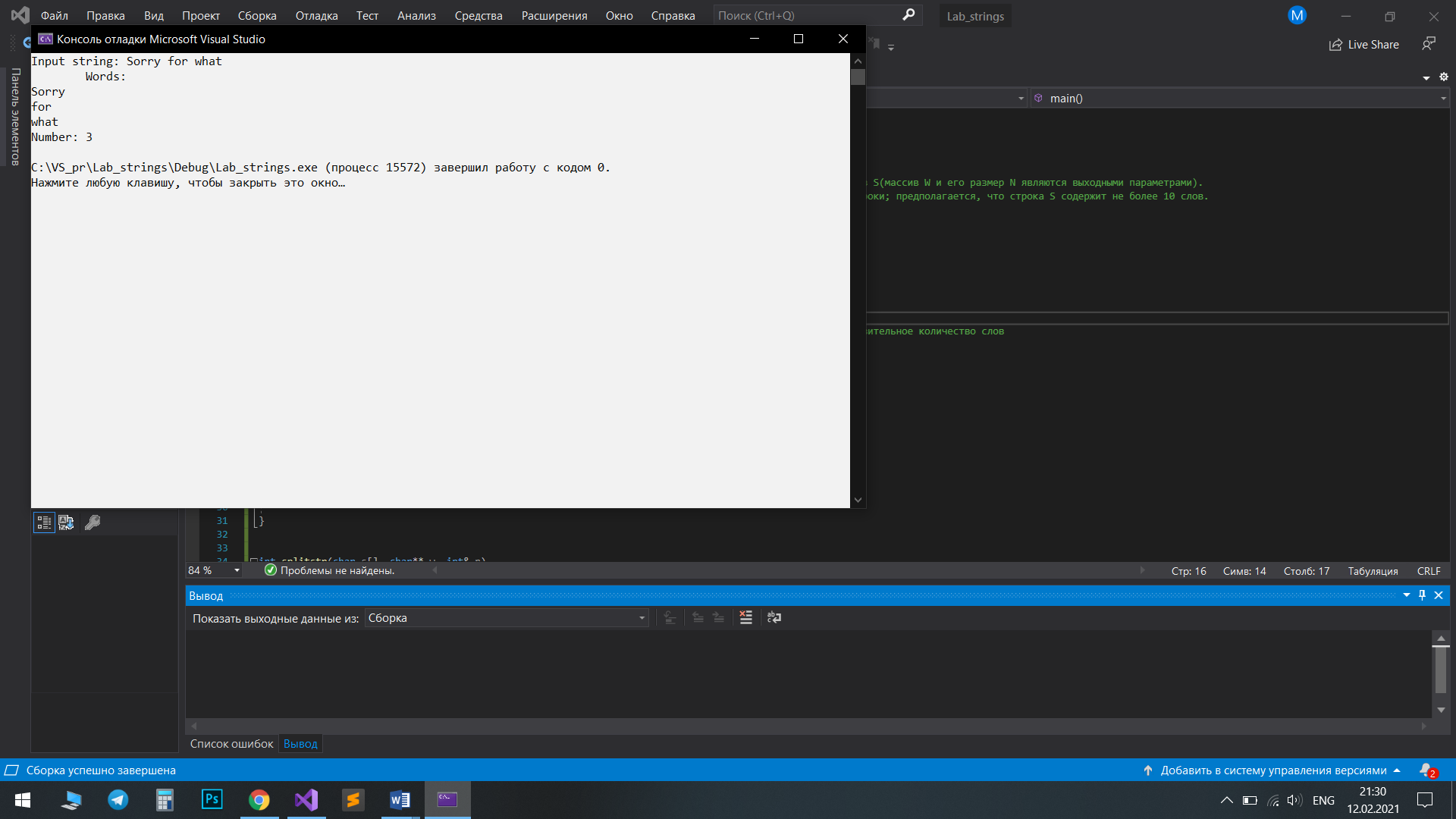
n += 1; //считаем количество слов

pch = strtok(NULL, " .\n");

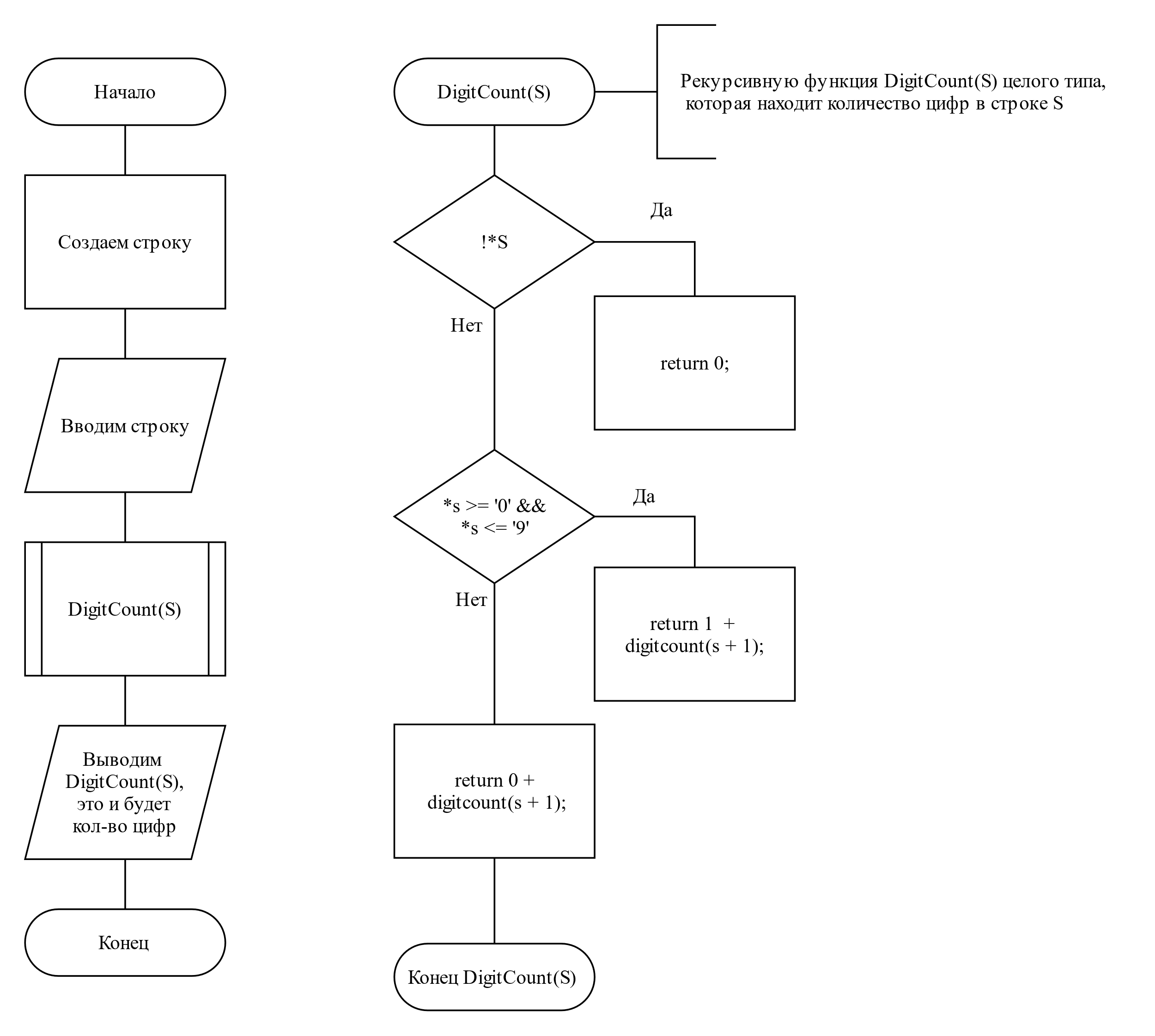
}

return n;} //возвращаем количество слов





**Задание 3.**



12. Описать рекурсивную функцию DigitCount(S) целого типа, которая находит количество цифр в строке S, не используя оператор цикла.

С помощью этой функции найти количество цифр в каждой из пяти данных строк.

#include<string.h>

#include<iostream>

#pragma warning(disable : 4996)

using namespace std;

int main(){

char string[20];

cout << "Enter string: ";

gets\_s(string);

cout<<digitcount(string)<<" :digits... "<<endl;

}

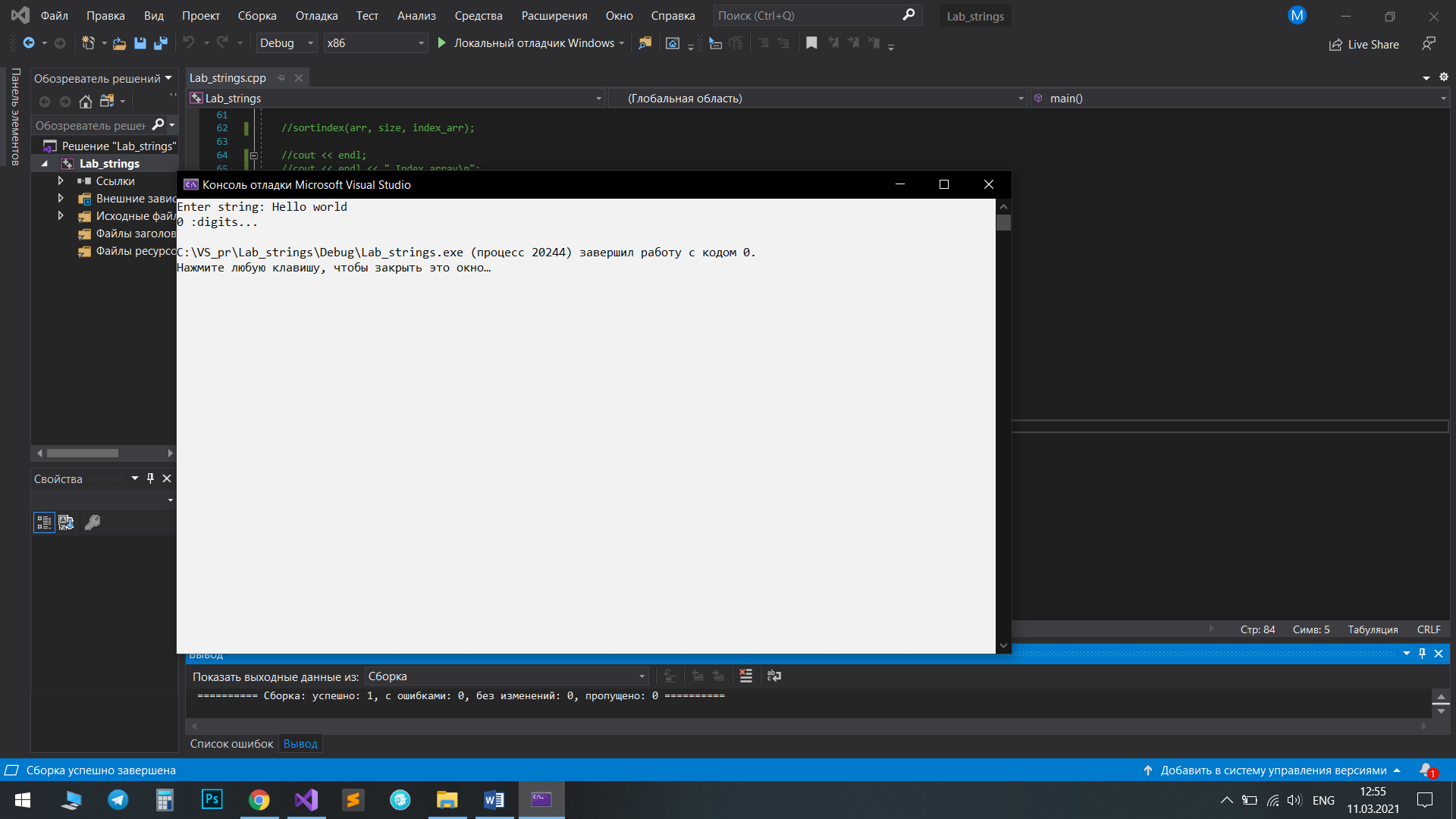
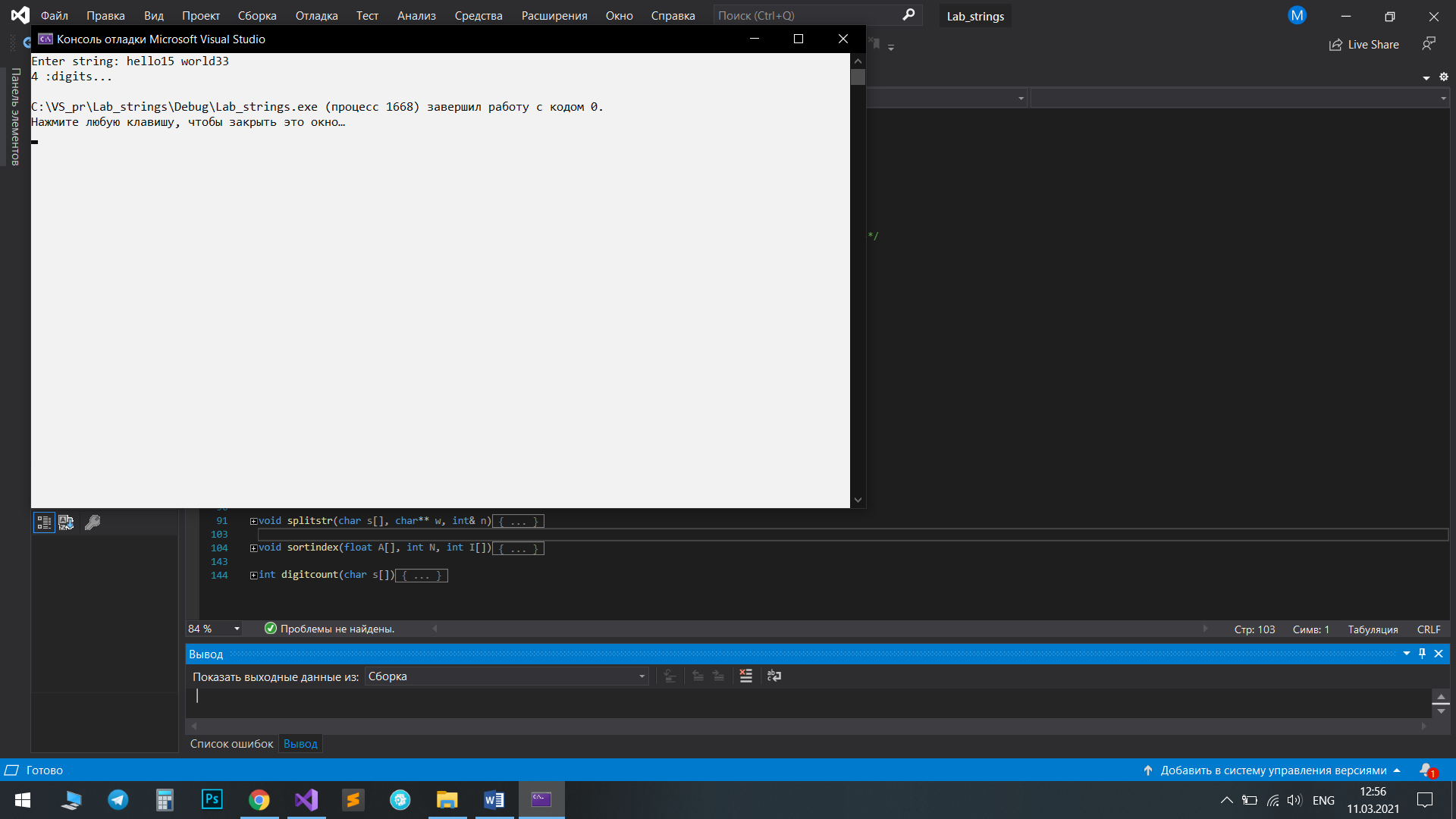
int digitcount(char s[])

{

if (!\*s) return 0;

return (\*s >= '0' && \*s <= '9' ? 1 : 0) + digitcount(s + 1);

}

Вывод: Я изучил основные принципы написания пользовательских функций, ознакомился с возможностями передачи данных в функции и получения результата по итогам работы функции. Реализовал собственные функции для обработки данных составных и простых типов.