Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.13**

Дисциплина: Основы теории алгоритмов и структуры данных.

Тема: “ Объектно-ориентированное программирование.

Стандартные обобщенные алгоритмы библиотеки STL”

**Вар.12**

Выполнила работу:

студент группы ИВТ-20-2Б

Ананина Арина Юрьевна

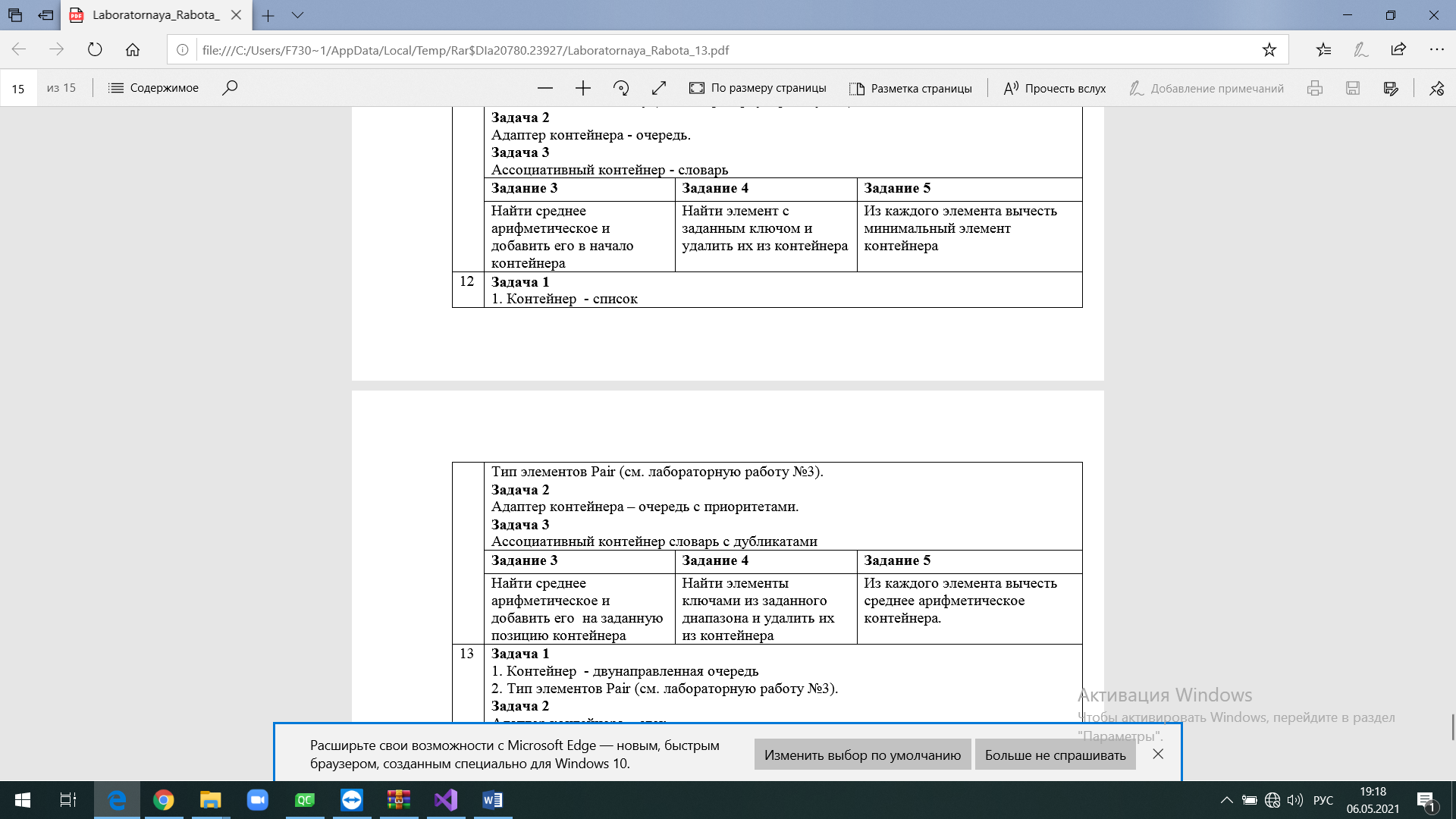
Проверила:

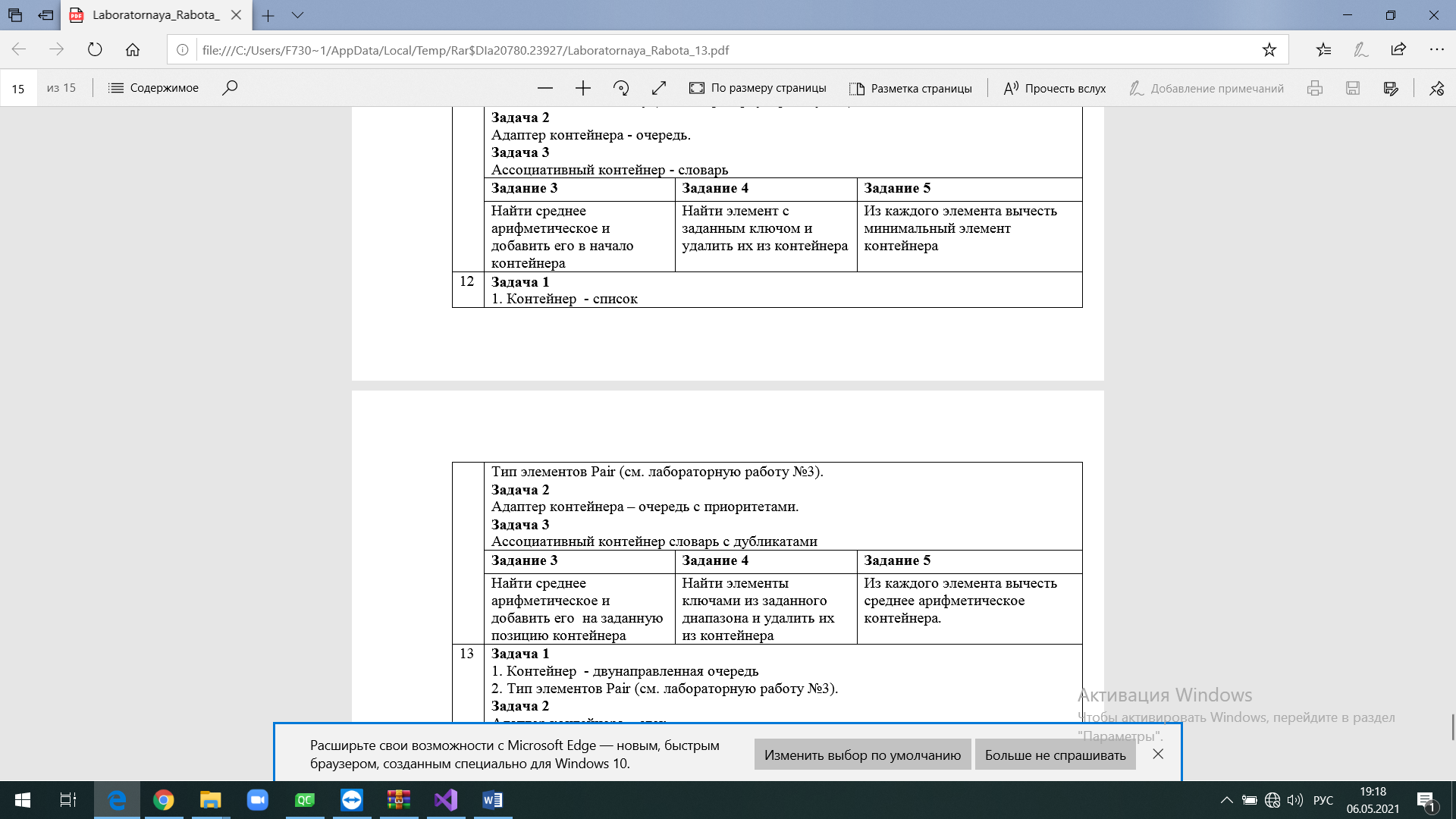
Доцент кафедры ИТАС

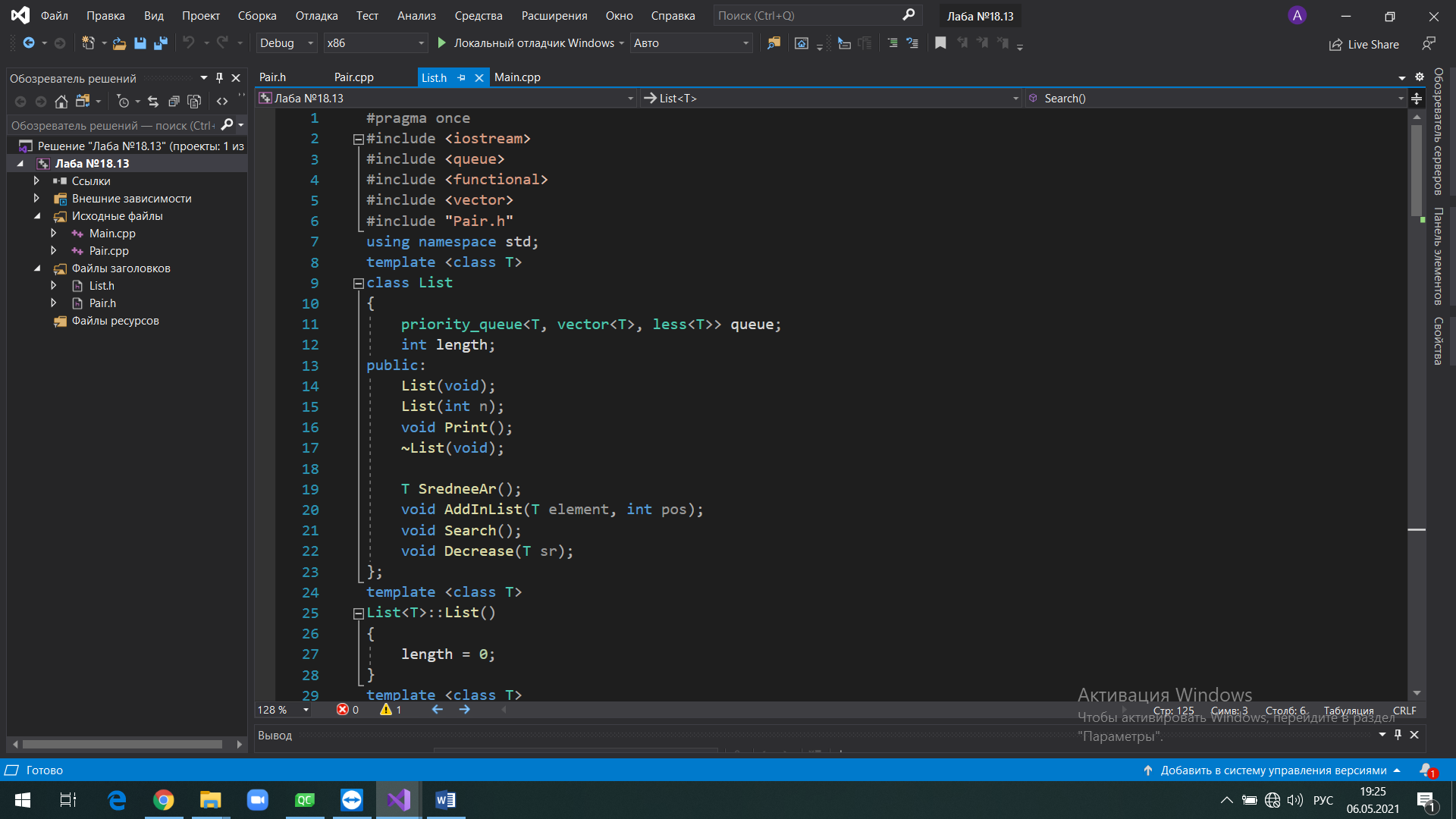
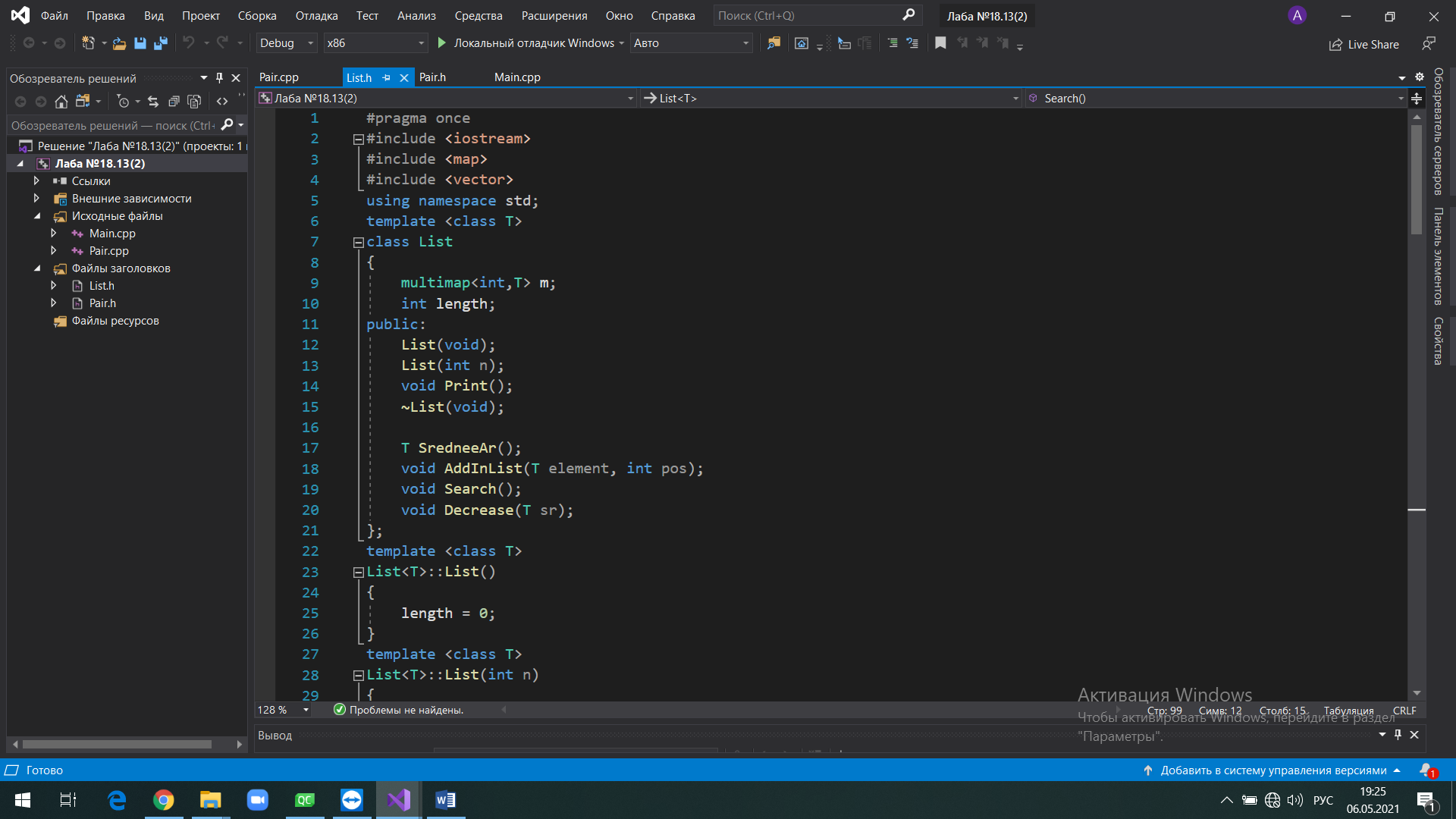
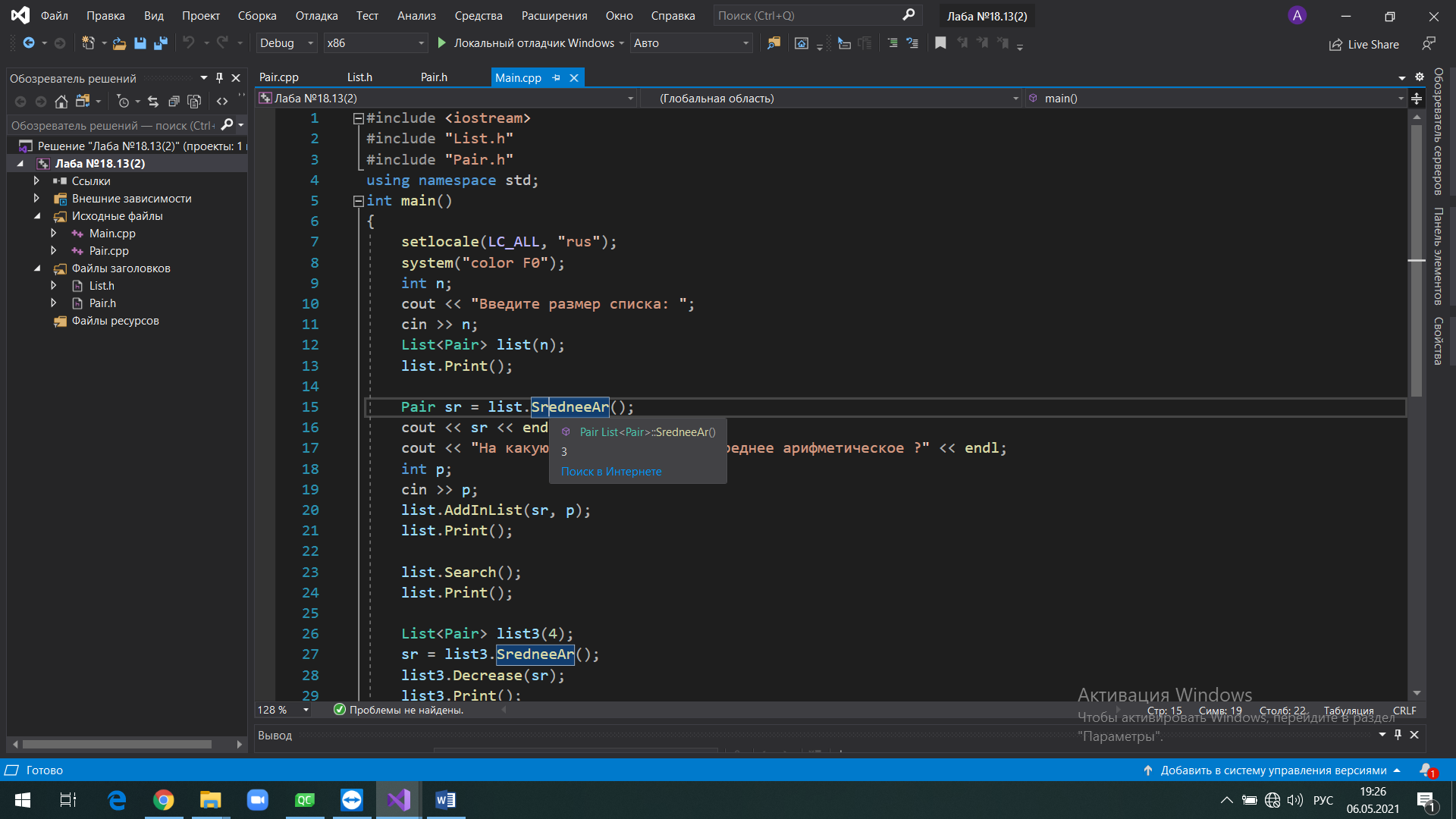
Полякова О.А.

Пермь, 2021

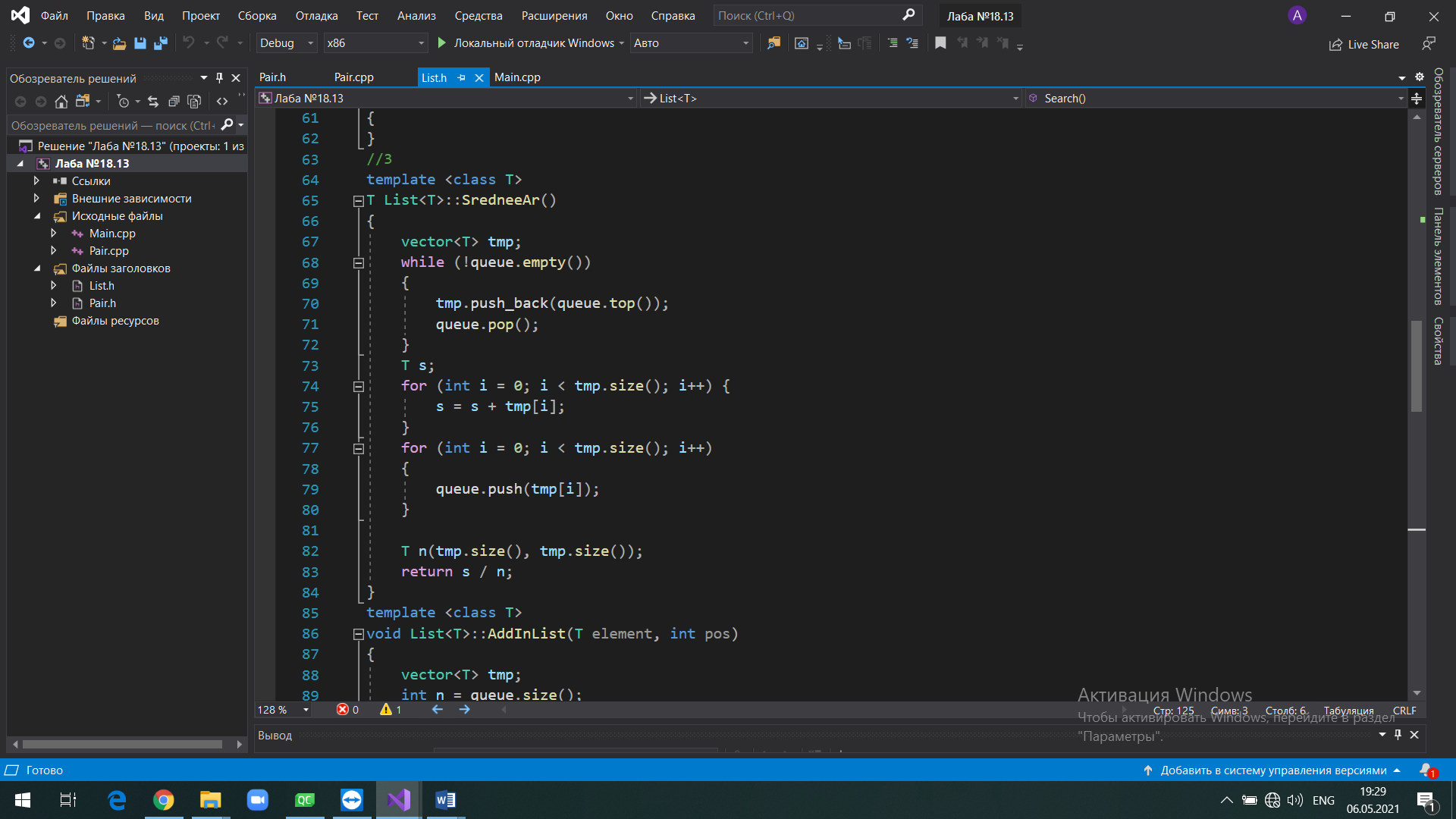
1. **Постановка задачи**



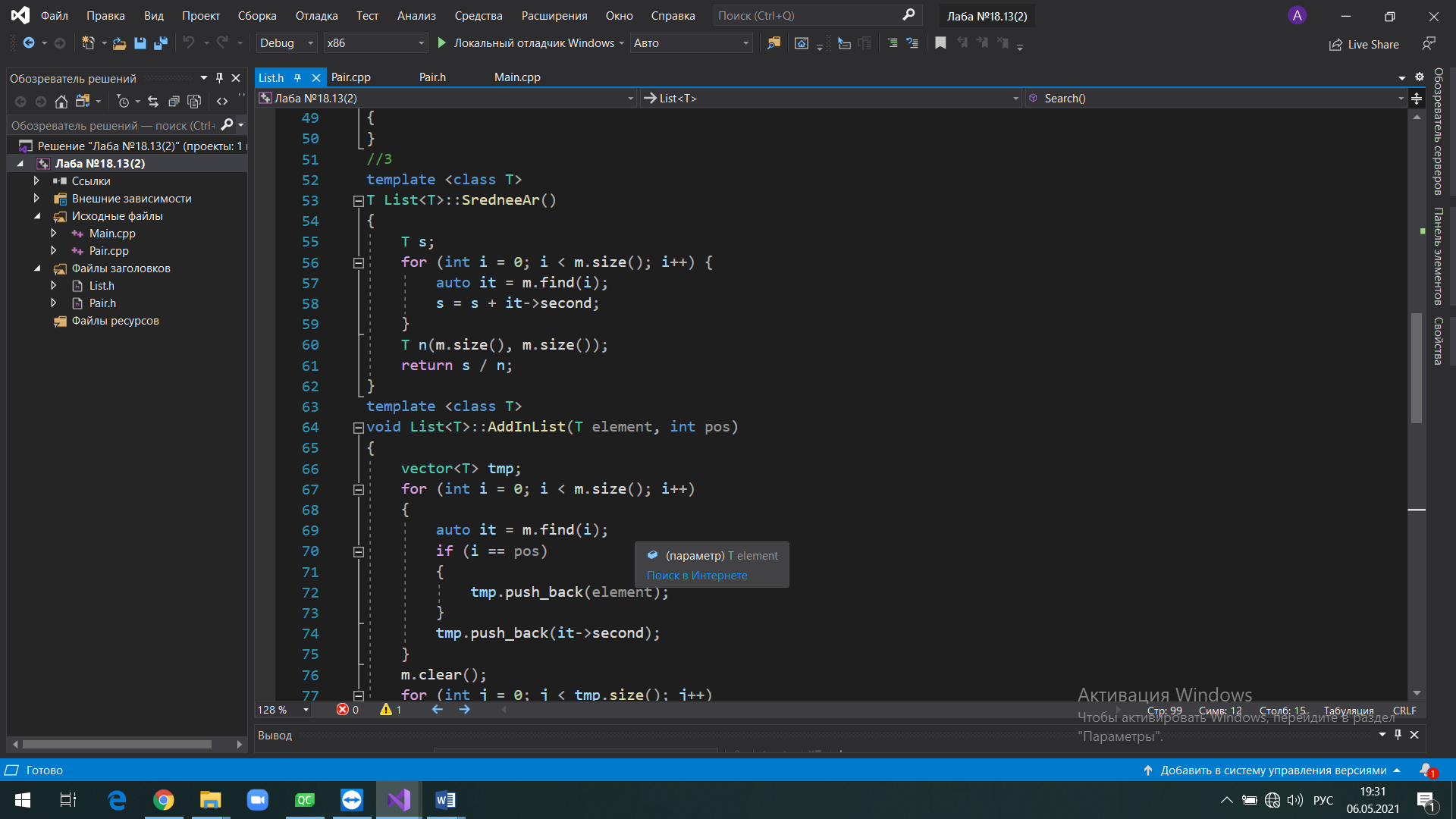


1. **Анализ задачи**
2. Для решения задачи необходимо:
   1. Использовать класс Pair.
   2. Использовать контейнеры библиотеки STL.
   3. Использовать алгоритмы библиотеки STL.
3. В ходе работы были использованы следующие типы данных:
   1. Pair, int, double, pair<int, double>.
4. Для решения задачи данные были представлены в следующем виде:
   1. 
   2. 
   3. 
5. Для операций ввода и вывода использовались следующие операторы и функции:
   1. Операторы cout и cin.
   2. Перегрузка операторов в классе Pair, функция Print и конструктор для класса List.
6. Поставленные задачи будут решены следующими действиями:
   1. С помощью функции SredneeAr будет найдено среднее арифметическое значение списка, а в функции AddInList будет организована вставка найденного значения на заданную позицию:

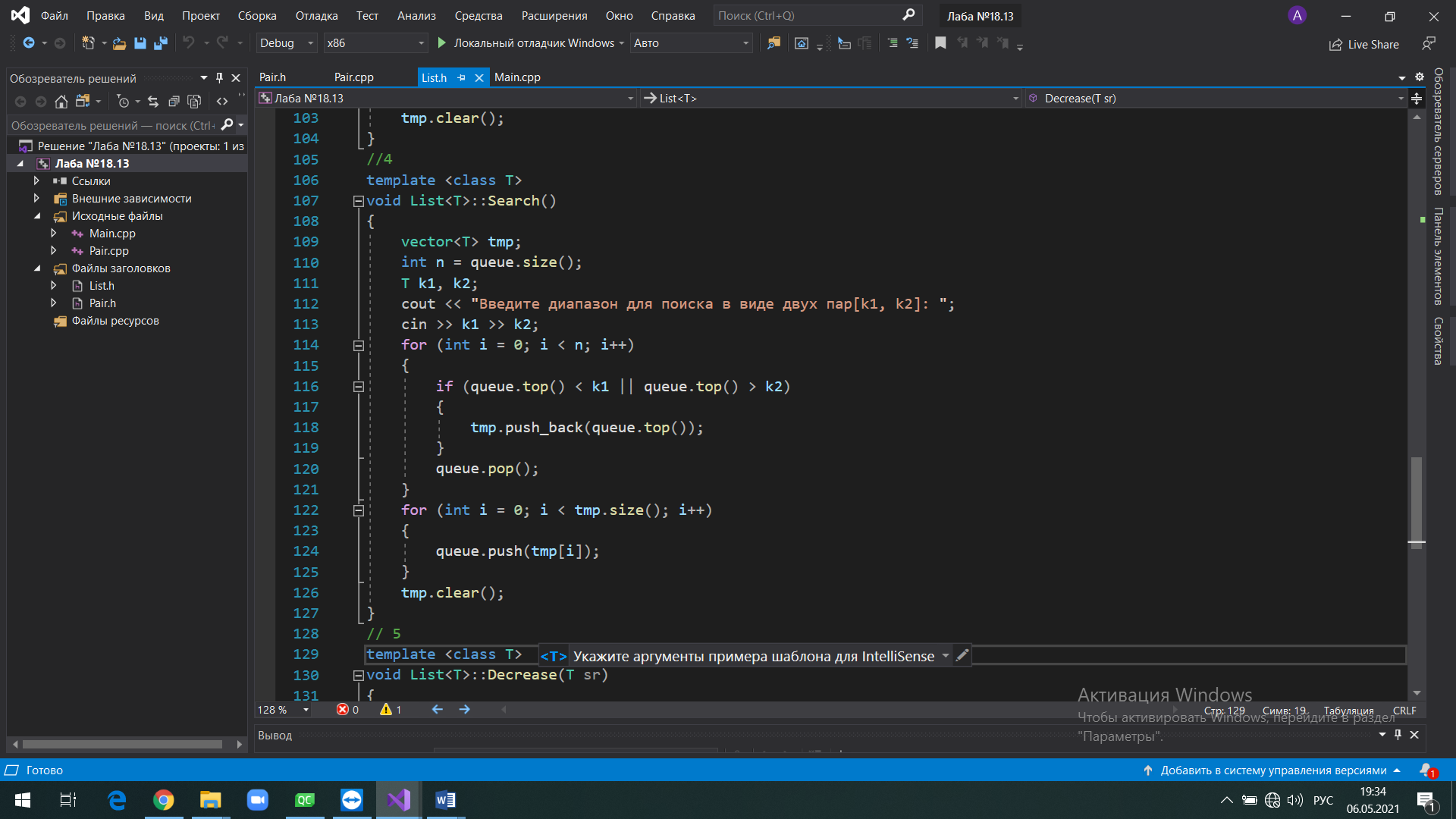
* Для задачи 2



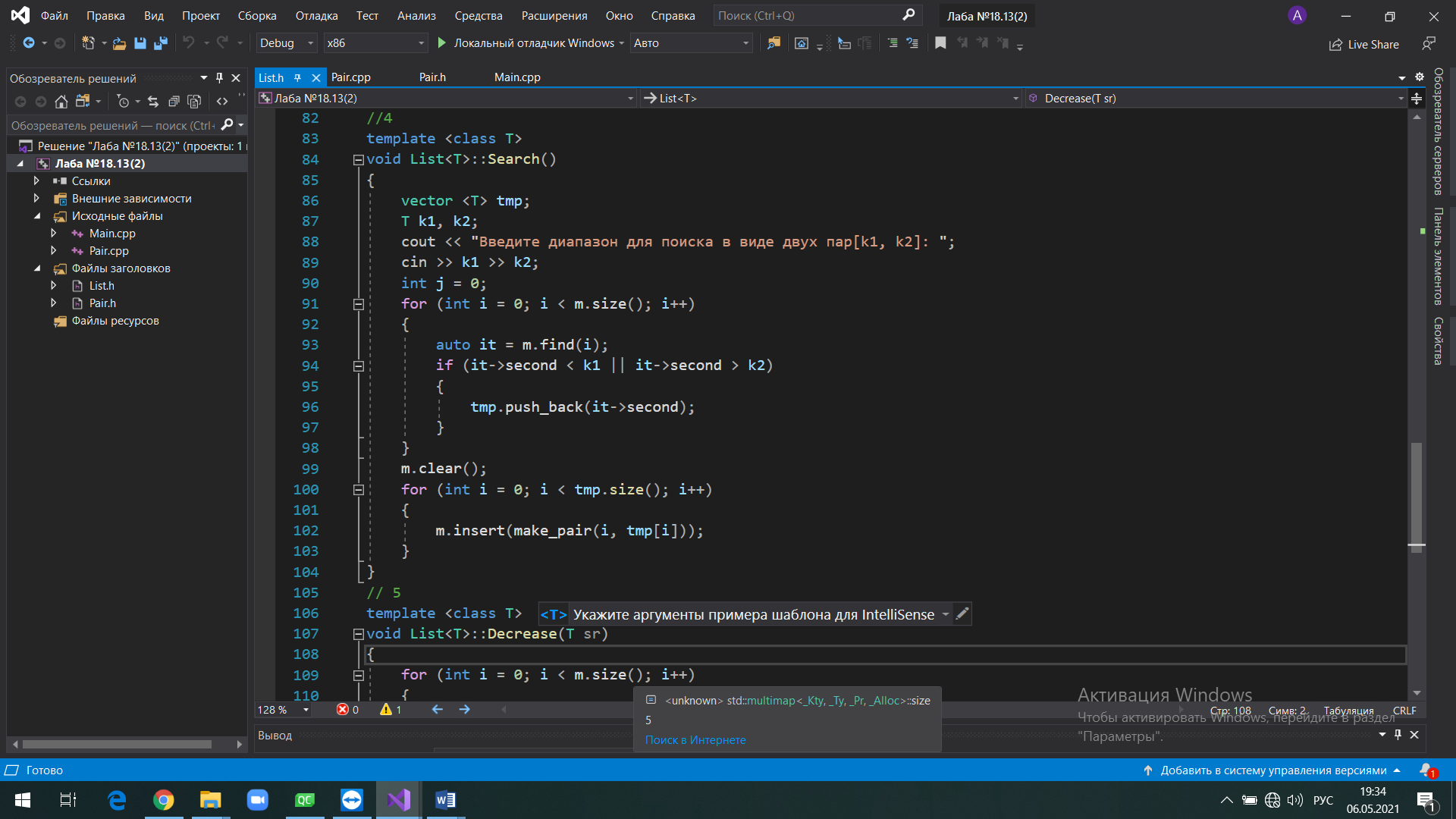
* Для задачи 3



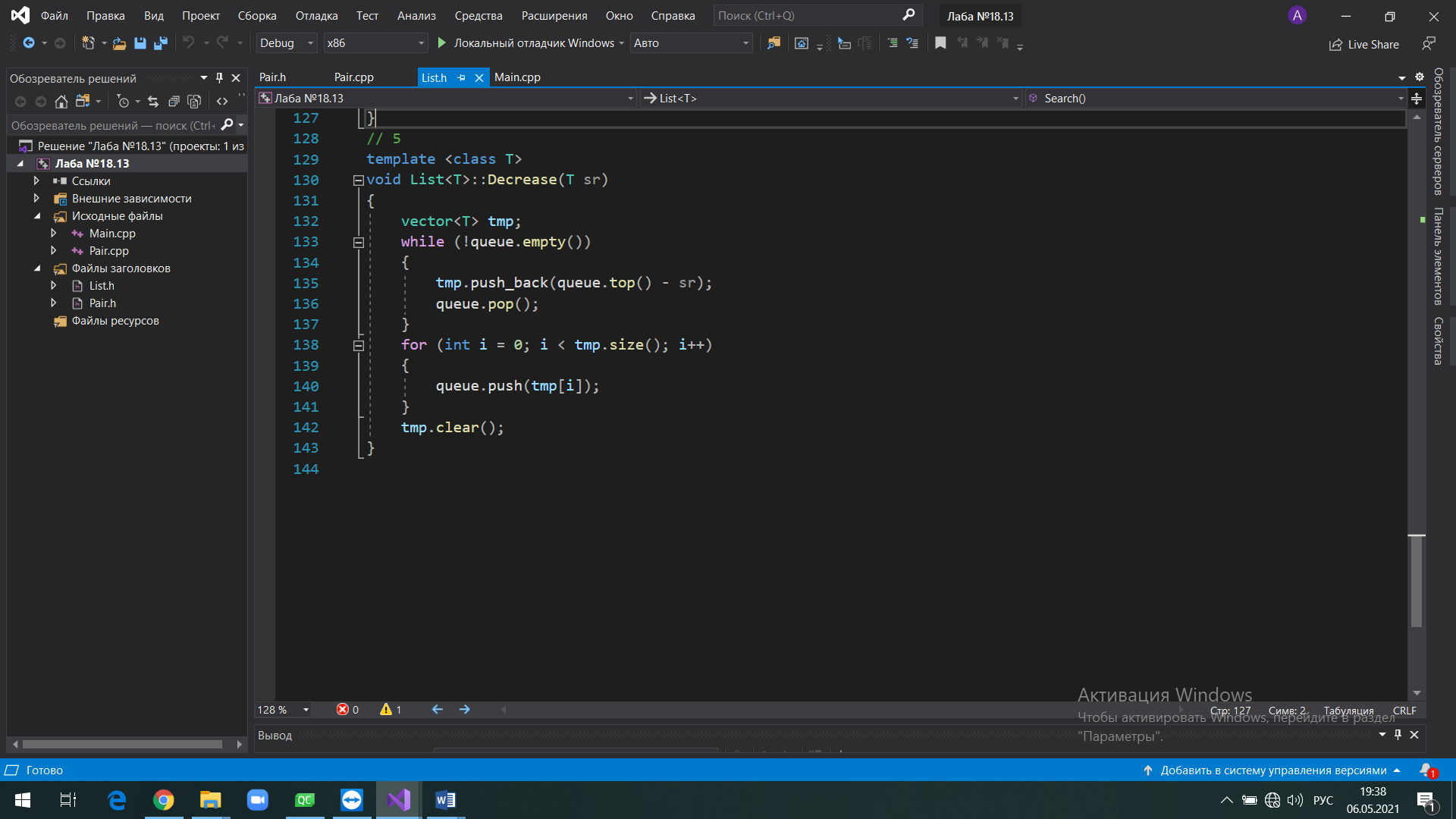
* 1. С помощью функции Search будет осуществлено удаление всех элементов из заданного диапазона чисел:
* Для задачи 2



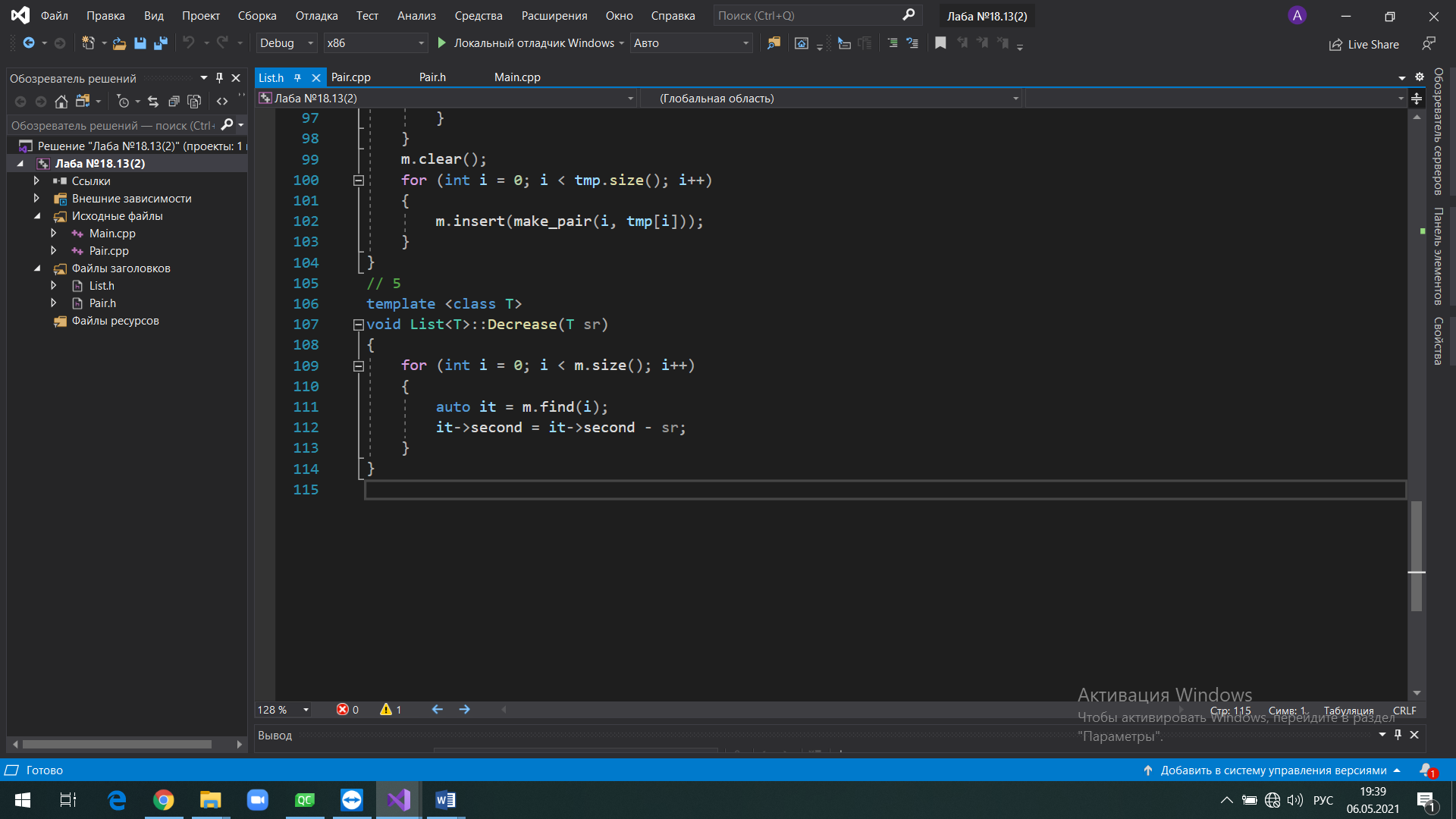
* Для задачи 3



* 1. С помощью функции Decrease все элементы списка будут уменьшены на среднее арифметическое этого списка:
* Для задачи 2



* Для задачи 3



1. **Код программы**

**Задача 2:**

Main.cpp:

#include <iostream>

#include "List.h"

#include "Pair.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

system("color F0");

int n;

cout << "Введите размер списка: ";

cin >> n;

List<Pair> list(n);

list.Print();

Pair sr = list.SredneeAr();

cout << sr << endl;

cout << "На какую позицию поставить среднее арифметическое ?" << endl;

int p;

cin >> p;

list.AddInList(sr, p);

list.Print();

list.Search();

list.Print();

List<Pair> list3(4);

sr = list3.SredneeAr();

list3.Decrease(sr);

list3.Print();

}

List.h:

#pragma once

#include <iostream>

#include <queue>

#include <functional>

#include <vector>

#include "Pair.h"

using namespace std;

template <class T>

class List

{

priority\_queue<T, vector<T>, less<T>> queue;

int length;

public:

List(void);

List(int n);

void Print();

~List(void);

T SredneeAr();

void AddInList(T element, int pos);

void Search();

void Decrease(T sr);

};

template <class T>

List<T>::List()

{

length = 0;

}

template <class T>

List<T>::List(int n)

{

T a;

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

{

cin >> a;

queue.push(a);

}

}

template <class T>

void List<T>::Print()

{

vector<T> tmp;

while (!queue.empty())

{

tmp.push\_back(queue.top());

queue.pop();

}

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

{

cout << i << " элемент: " << tmp[i] << endl;

}

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

{

queue.push(tmp[i]);

}

tmp.clear();

cout << endl;

}

template <class T>

List<T>::~List()

{

}

//3

template <class T>

T List<T>::SredneeAr()

{

vector<T> tmp;

while (!queue.empty())

{

tmp.push\_back(queue.top());

queue.pop();

}

T s;

for (int i = 0; i < tmp.size(); i++) {

s = s + tmp[i];

}

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

{

queue.push(tmp[i]);

}

T n(tmp.size(), tmp.size());

return s / n;

}

template <class T>

void List<T>::AddInList(T element, int pos)

{

vector<T> tmp;

int n = queue.size();

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

{

if (i == pos)

{

tmp.push\_back(element);

}

tmp.push\_back(queue.top());

queue.pop();

}

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

{

queue.push(tmp[i]);

}

tmp.clear();

}

//4

template <class T>

void List<T>::Search()

{

vector<T> tmp;

int n = queue.size();

T k1, k2;

cout << "Введите диапазон для поиска в виде двух пар[k1, k2]: ";

cin >> k1 >> k2;

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

{

if (queue.top() < k1 || queue.top() > k2)

{

tmp.push\_back(queue.top());

}

queue.pop();

}

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

{

queue.push(tmp[i]);

}

tmp.clear();

}

// 5

template <class T>

void List<T>::Decrease(T sr)

{

vector<T> tmp;

while (!queue.empty())

{

tmp.push\_back(queue.top() - sr);

queue.pop();

}

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

{

queue.push(tmp[i]);

}

tmp.clear();

}

**Задача 3:**

Main.cpp

#include <iostream>

#include "List.h"

#include "Pair.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

system("color F0");

int n;

cout << "Введите размер списка: ";

cin >> n;

List<Pair> list(n);

list.Print();

Pair sr = list.SredneeAr();

cout << sr << endl;

cout << "На какую позицию поставить среднее арифметическое ?" << endl;

int p;

cin >> p;

list.AddInList(sr, p);

list.Print();

list.Search();

list.Print();

List<Pair> list3(4);

sr = list3.SredneeAr();

list3.Decrease(sr);

list3.Print();

}

List.h

#pragma once

#include <iostream>

#include <map>

#include <vector>

using namespace std;

template <class T>

class List

{

multimap<int,T> m;

int length;

public:

List(void);

List(int n);

void Print();

~List(void);

T SredneeAr();

void AddInList(T element, int pos);

void Search();

void Decrease(T sr);

};

template <class T>

List<T>::List()

{

length = 0;

}

template <class T>

List<T>::List(int n)

{

T a;

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

{

cin >> a;

m.insert(make\_pair(i, a));

}

}

template <class T>

void List<T>::Print()

{

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

{

auto it = m.find(i);

cout << i << " элемент: " << it->second << endl;

}

cout << endl;

}

template <class T>

List<T>::~List()

{

}

//3

template <class T>

T List<T>::SredneeAr()

{

T s;

for (int i = 0; i < m.size(); i++) {

auto it = m.find(i);

s = s + it->second;

}

T n(m.size(), m.size());

return s / n;

}

template <class T>

void List<T>::AddInList(T element, int pos)

{

vector<T> tmp;

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

{

auto it = m.find(i);

if (i == pos)

{

tmp.push\_back(element);

}

tmp.push\_back(it->second);

}

m.clear();

for (int j = 0; j < tmp.size(); j++)

{

m.insert(make\_pair(j, tmp[j]));

}

}

//4

template <class T>

void List<T>::Search()

{

vector <T> tmp;

T k1, k2;

cout << "Введите диапазон для поиска в виде двух пар[k1, k2]: ";

cin >> k1 >> k2;

int j = 0;

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

{

auto it = m.find(i);

if (it->second < k1 || it->second > k2)

{

tmp.push\_back(it->second);

}

}

m.clear();

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

{

m.insert(make\_pair(i, tmp[i]));

}

}

// 5

template <class T>

void List<T>::Decrease(T sr)

{

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

{

auto it = m.find(i);

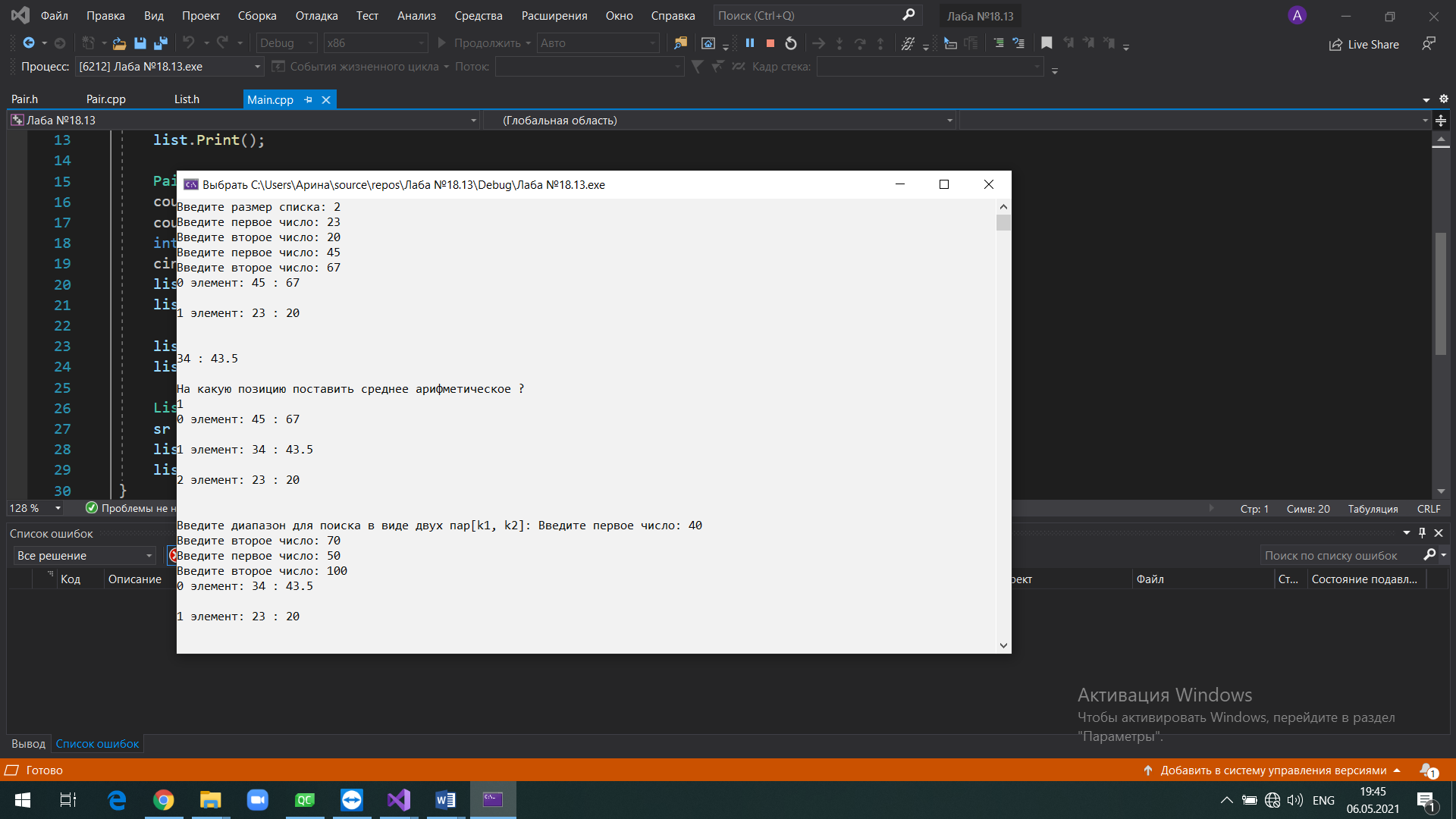
it->second = it->second - sr;

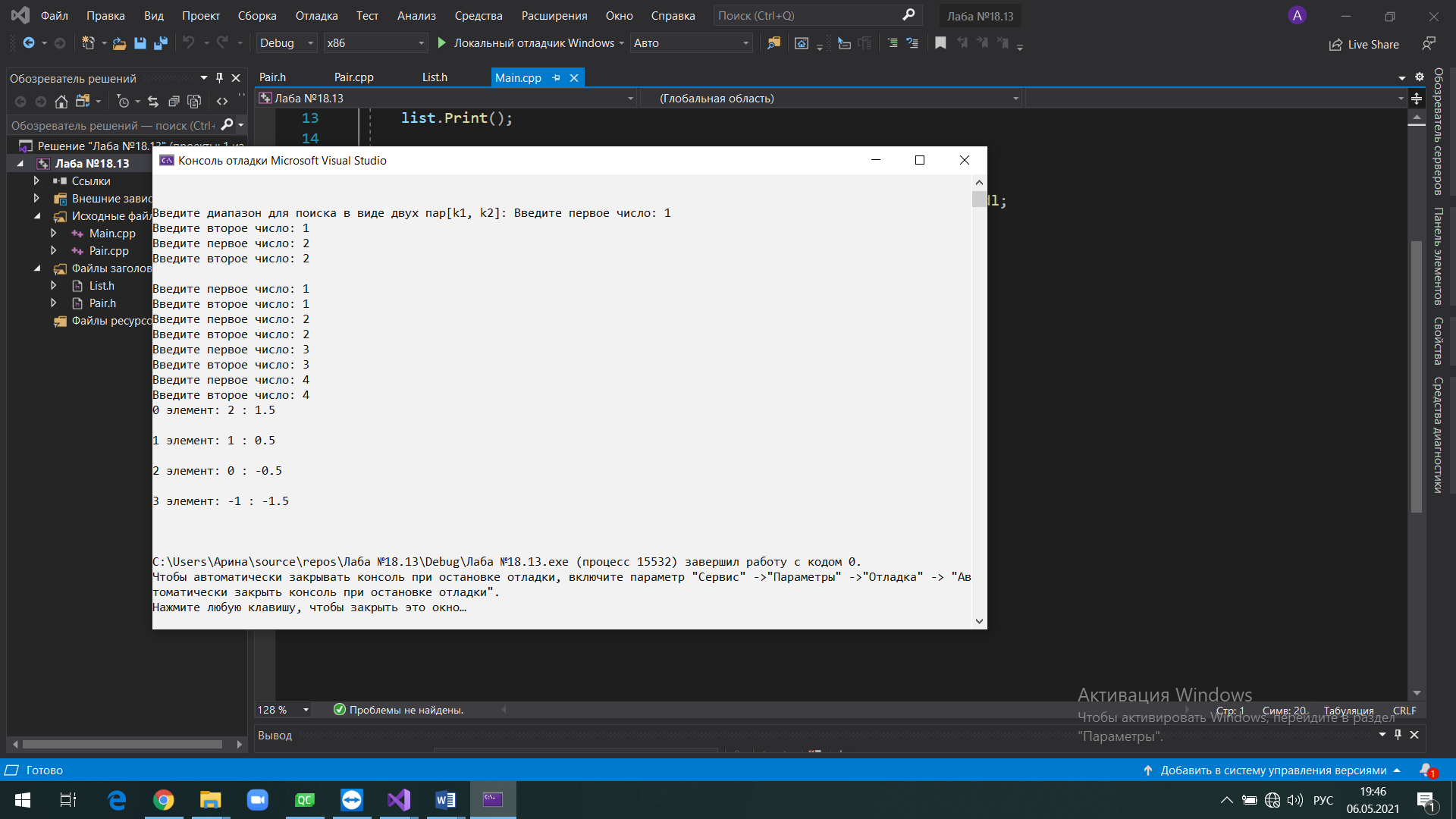
}

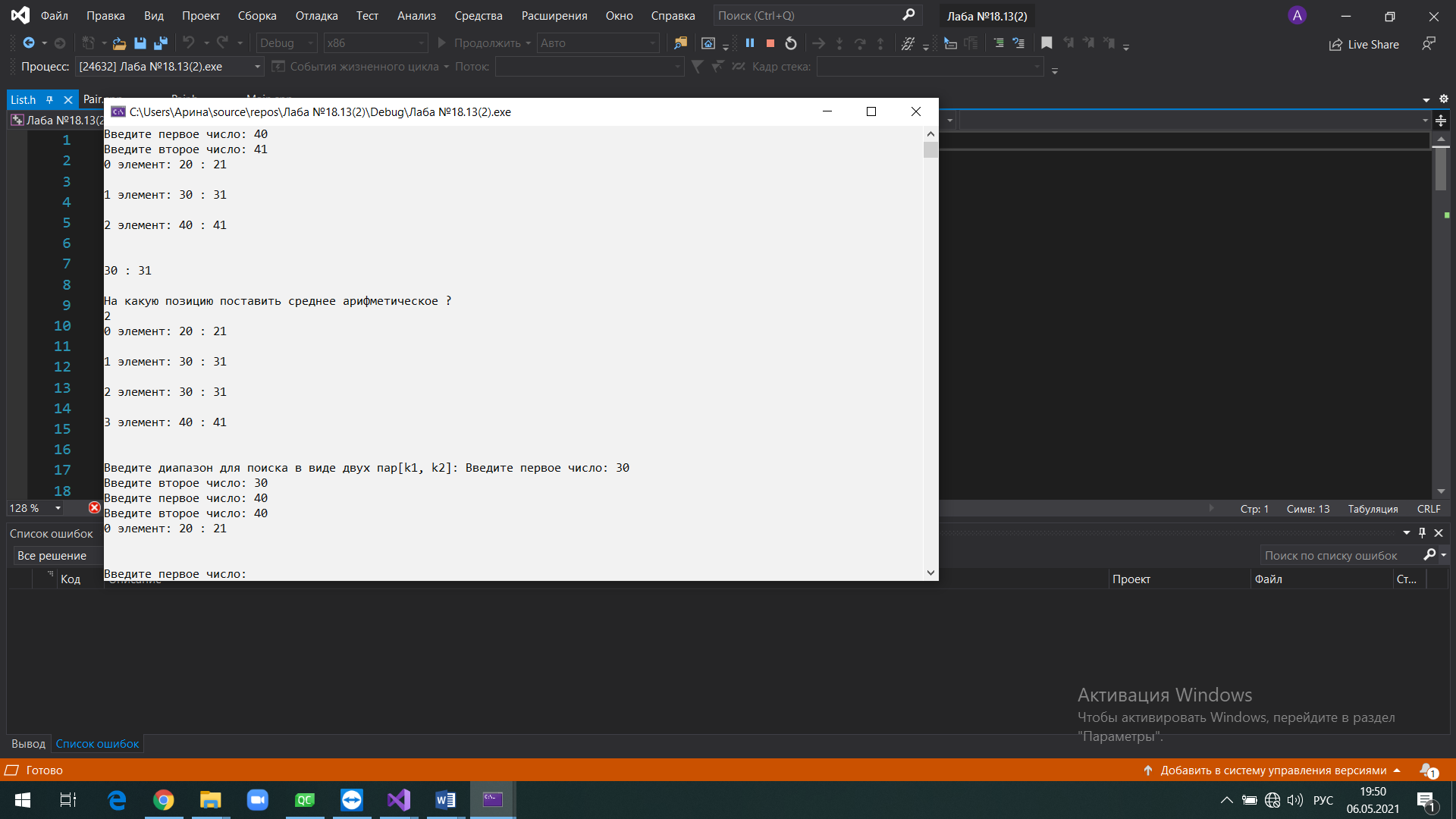
}

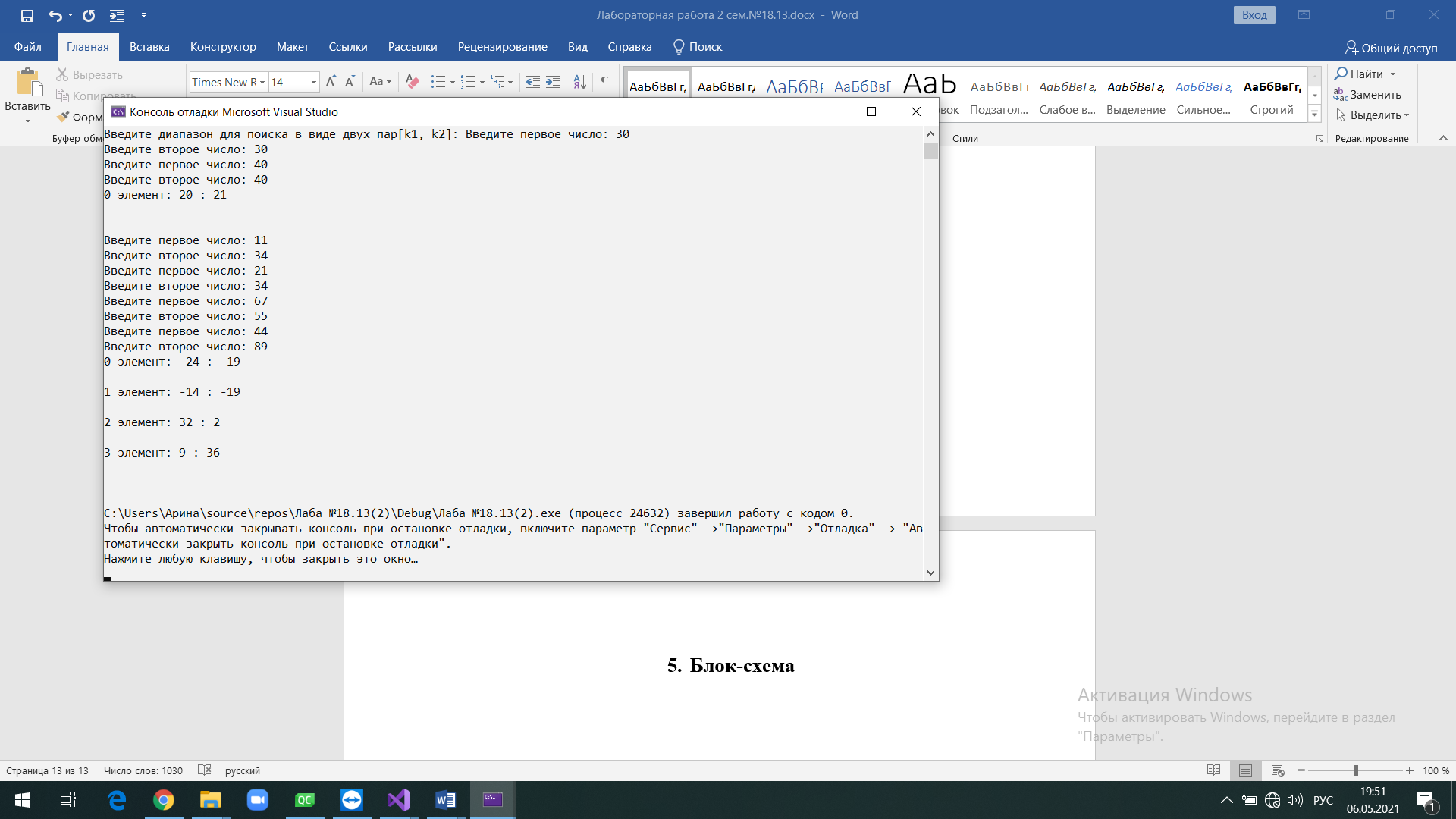
1. **Выполнение программы**

Задача 2:





Задача 3:  




1. **Блок-схема**

