MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

NATIONAL TECHNICAL UNIVERSITY

“KHARKOV POLYTECHNICAL INSTITUTE”

LABORATORY WORK № 2

C++ Operators and Statements

Created by student of 1.КН201.8г

Pasichnuy O.S.

Checked by

Lutenko I.V.

KHARKIV 2018

Topic: C++ Operators and Statements

Goal: Learning basic principles of C++

Tasks:

1.Develop a program that implements an algorithm for solving quadratic equation. The algorithm should consider all possible data.

2.Develop a program that implements an algorithm for calculating the following expression. Provide a check of possible errors.

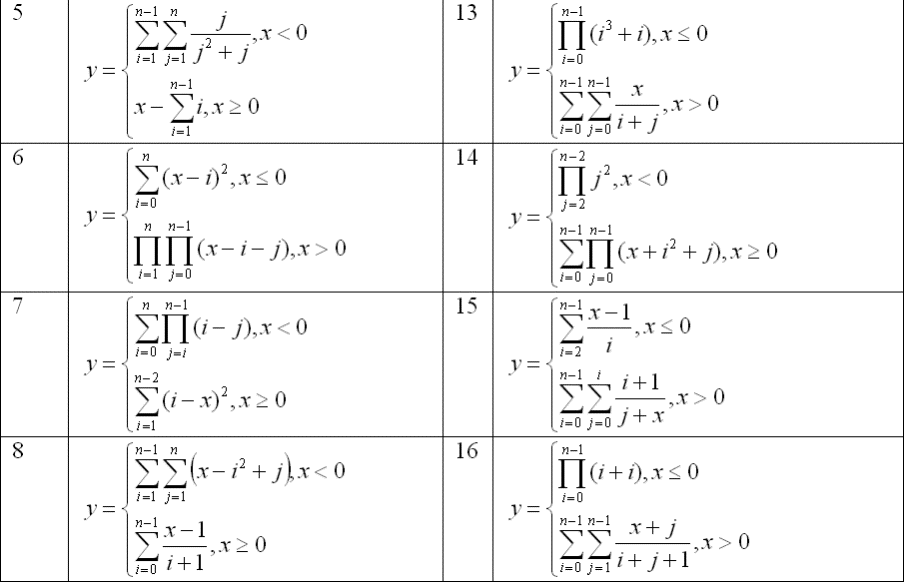
y = 1/(x + 2) + 2/(x + 4) + ... + (k - 1)/(x + 2(k - 1)) + (k + 1)/(x + 2(k + 1)) + ... + n/(x + 2n)

3.Write a program that reads x and n and calculates y:  
 y = (x + 1)(x - 2)(x + 3)(x - 4) ... (x - 2n)

4.Write a program that reads eps and calculates y:

y = 1/2 + 1/4 + 1/8 + 1/16 + ...  
The loop terminates if new summand is less than eps

5.Develop the code for the task from previous lab.



**Task 1**

class SquareEquation //maybe done

{

private:

double a, b, c, d, x1, x2;

public:

void findResult()

{

input("Input your 'a': ", a);

input("Input your 'b': ", b);

input("Input your 'c': ", c);

d = b\*b - 4 \* a\*c;

if (d < 0)

{

std::cout << "No roots, negative D!!!\n";

return;

}

x1 = (-b + sqrt(d))/2\*a;

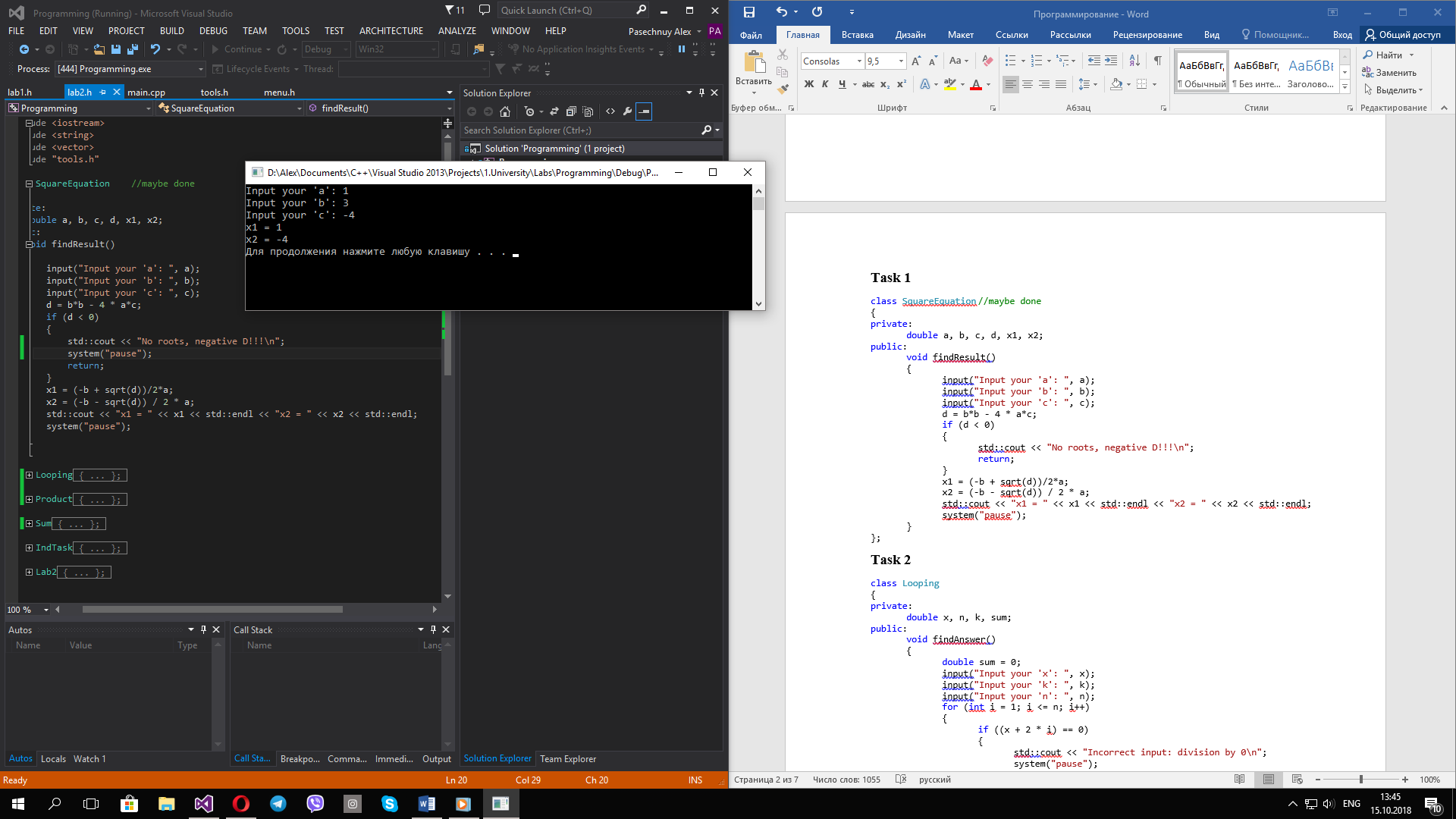
x2 = (-b - sqrt(d)) / 2 \* a;

std::cout << "x1 = " << x1 << std::endl << "x2 = " << x2 << std::endl;

system("pause");

}

};



**Task 2**

class Looping

{

private:

double x, n, k, sum;

public:

void findAnswer()

{

double sum = 0;

input("Input your 'x': ", x);

input("Input your 'k': ", k);

input("Input your 'n': ", n);

for (int i = 1; i <= n; i++)

{

if ((x + 2 \* i) == 0)

{

std::cout << "Incorrect input: division by 0\n";

system("pause");

return;

}

if(i != k)

sum += 1 / (x + 2 \* i);

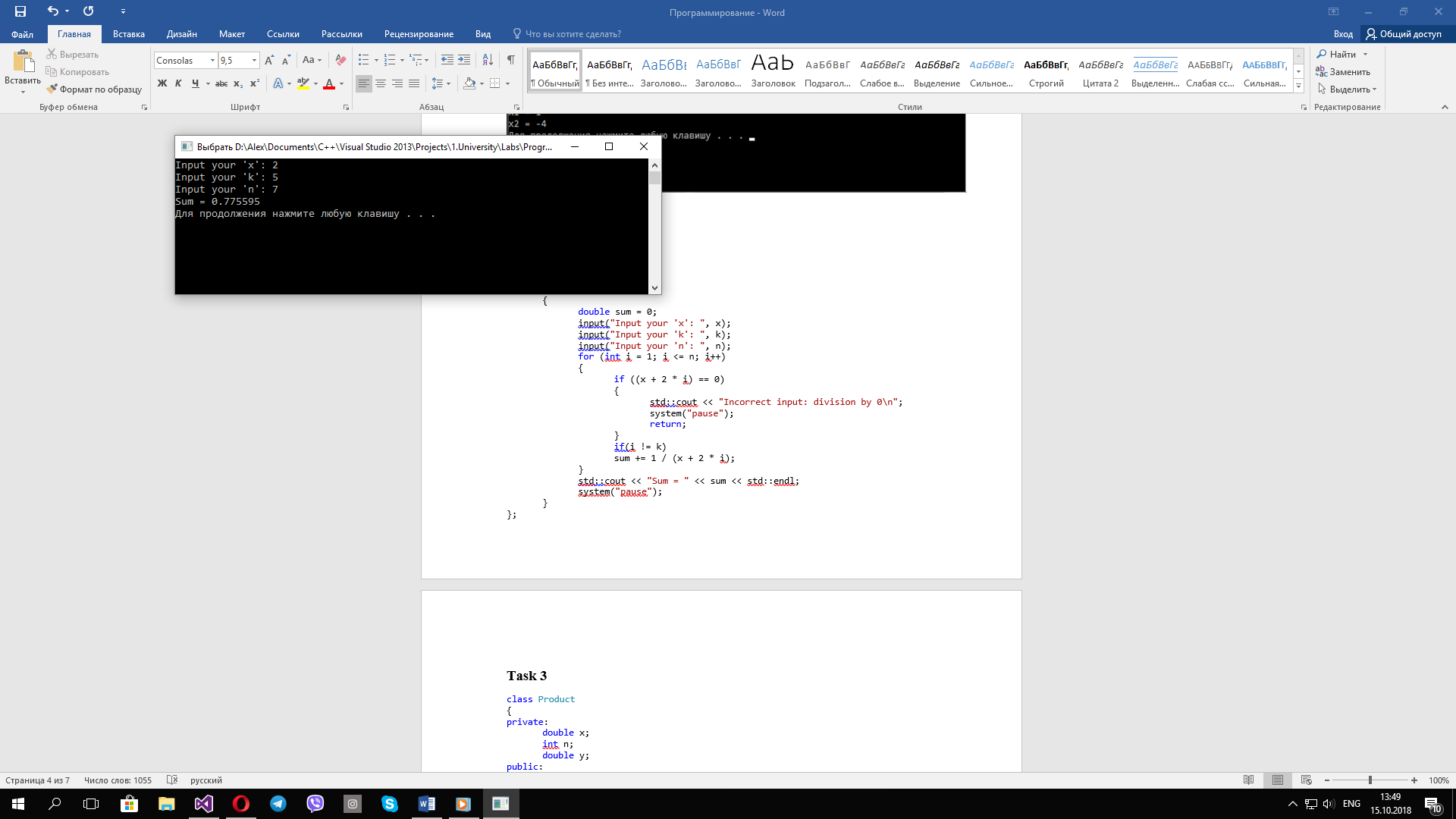
}

std::cout << "Sum = " << sum << std::endl;

system("pause");

}

};



**Task 3**

class Product

{

private:

double x;

int n;

double y;

public:

void calculate()

{

int minuser = 1;

y = 1;

input("Input your 'x': ", x);

while (true)

{

input("Input your 'n': ", n);

if (n > 0)

break;

std::cout << "Your 'n' is too small, please try again!" << std::endl;

}

for (int i = 1; i < 2 \* n; i++)

{

y \*= (x + (minuser \* i));

minuser \*= -1;

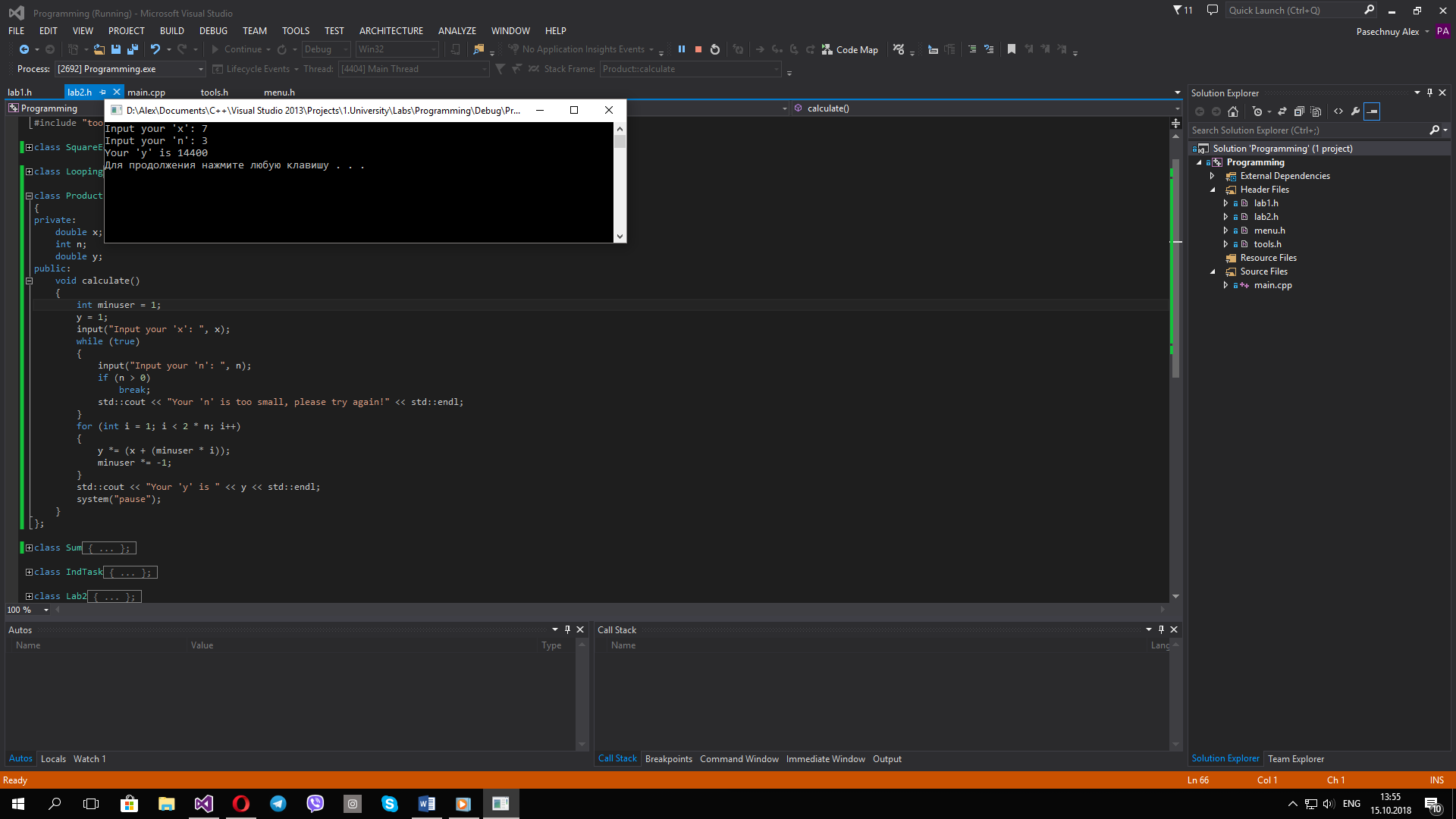
}

std::cout << "Your 'y' is " << y << std::endl;

system("pause");

}

};



**Task 4**

class Sum

{

private:

double sum;

double eps;

public:

void calculate()

{

sum = 0;

while (true)

{

input("Enter your 'eps': ", eps);

if (eps < 0.5 && eps > 0)

break;

std::cout << "Incorrect 'eps'!" << std::endl;

}

double invEps = 1 / eps;

for (int i = 2; i < invEps; i \*= 2)

{

sum += 1.0 / i;

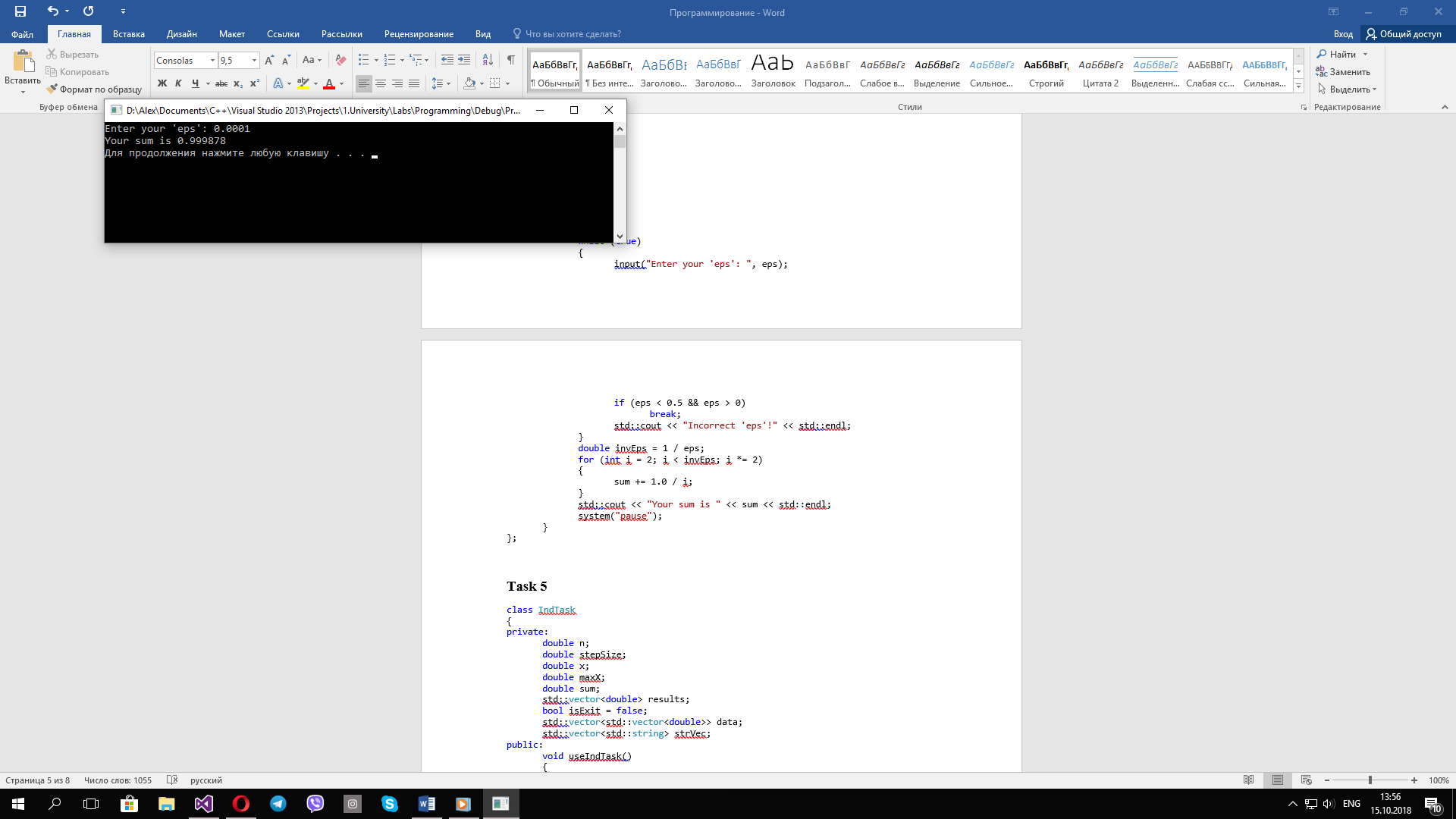
}

std::cout << "Your sum is " << sum << std::endl;

system("pause");

}

};



**Task 5**

class IndTask

{

private:

double n;

double stepSize;

double x;

double maxX;

double sum;

std::vector<double> results;

bool isExit = false;

std::vector<std::vector<double>> data;

std::vector<std::string> strVec;

public:

void useIndTask()

{

while (isExit == false)

{

getData();

for (x; x <= maxX; x += stepSize)

{

if (x >= 0)

positiveX();

else

negativeX();

std::cout << "x = " << x << "; sum = " << sum << std::endl;

results.push\_back(sum);

}

returnToFile();

system("cls");

std::cout << "Do you want to use the program again?\n";

switch (Menu::getInstance().menuOrgan({ "Yes", "No" }))

{

case 0:

break;

case 1:

isExit = true;

std::cout << "Thanks for using my product)\n"

<< "With love, Alex Pasechnuy\n";

break;

}

}

}

void getData()

{

std::cout << "Select data input way please:\n";

std::vector<std::string> giveMenu = { "Give from file", "Give from keyboard" };

switch (Menu::getInstance().menuOrgan(giveMenu))

{

case 0:

fileWork();

break;

case 1:

getFromKeyboard();

break;

}

system("cls");

}

void fileWork()

{

getFromFile();

while (true)

{

system("cls");

std::cout << "Select your start and maximum value of x ,n and step size:\n";

int choise = Menu::getInstance().menuOrgan(strVec);

std::vector<double> vecDob = data[choise];

x = vecDob[0]; //interval start

maxX = vecDob[1]; //interval end

n = vecDob[2];

stepSize = vecDob[3];

int minN;

if (x < 0)

minN = 2;

else

minN = 1;

if (maxX >= x && stepSize > 0 && n >= minN)

return;

if (maxX < x)

std::cout << "!Your interval has negative direction!" << std::endl;

if (stepSize <= 0)

std::cout << "!Incorrect step size!" << std::endl;

if (n < minN)

std::cout << "!Your n is too small for this x!" << std::endl;

system("pause");

}

}

void getFromFile()

{

std::ifstream fin;

fin.open("InputLab2.txt");

int memb;

while (!fin.eof())

{

std::string str;

std::vector<double> vecDob;

for (int i = 0; i < 4; i++)

{

fin >> memb;

vecDob.push\_back(memb);

str += std::to\_string(memb) + " ";

}

strVec.push\_back(str);

data.push\_back(vecDob);

}

fin.close();

}

void getFromKeyboard()

{

while (true)

{

input("start value of your x :", x);

input("maximum value of your x :", maxX);

if (x <= maxX)

break;

std::cout << "Incorrect input: interval must have positive direction!" << std::endl;

}

while (true)

{

input("step = ", stepSize);

if (stepSize > 0)

break;

std::cout << "Incorrect input: step must be greater than 0!" << std::endl;

}

int minN;

if (x < 0)

minN = 2;

else

minN = 1;

while (true)

{

input("n = ", n);

if (n >= minN)

break;

std::cout << "This n is too small for your x!" << std::endl;

}

}

void positiveX()

{

sum = 0;

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

{

sum += (x - 1) / (i + 1);

}

}

void negativeX()

{

sum = 0;

for (int i = 1; i < n; i++)

{

for (int j = 1; j <= n; j++)

{

sum += x - (i\*i) + j;

}

}

}

void returnToFile()

{

std::ofstream fout;

fout.open("OutputLab2.txt", std::ios\_base::app);

std::cout << std::endl << "Do you want to save your results to file?" << std::endl;

switch (Menu::getInstance().menuOrgan({ "Yes", "No" }))

{

case 0:

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

{

fout << results[i] << " ";

}

fout << std::endl;

fout.close();

break;

case 1:

break;

}

}

};

