УО «Белорусский государственный университет информатики и

радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе № 3.2

по предмету

Основы Алгоритмизации и Программирования

Вариант 4

Выполнил

Воривода М.А.

Проверила

Данилова Г.В.

Группа:

951007

Минск 2019

**Задание**

Дана непустая последовательность символов, требуется построить и напечатать множество, элементами которого являются встречающиеся в последовательности знаки арифметических операций и числа.

**Код программы**

**(Delphi)**

program Goal2;

{$APPTYPE CONSOLE}

{$R \*.res}

uses

System.SysUtils;

type

SetOfChar = Set of Char;

const

PATTERN = ['0' .. '9', '.', '^', '!', '+', '-', '\*', '/'];

function Intersection(S: SetOfChar): SetOfChar;

begin

Intersection := S \* PATTERN;

end;

procedure InputData(var S: SetOfChar);

var

Valid: Boolean;

Reader: AnsiString;

I: Integer;

begin

Valid := false;

repeat

try

WriteLn('Enter set (set length should be less than

256): ');

ReadLn(Reader);

if Length(Reader) < 256 then

begin

Valid := true;

for I := 1 to Length(Reader) do

include(S, Reader[I]);

end;

except

WriteLn('Entered set is too large');

end;

until Valid;

end;

procedure InputDataFromFile(var S: SetOfChar);

var

InF: TextFile;

Valid: Boolean;

Reader: AnsiString;

I: Integer;

begin

Valid := false;

repeat

try

WriteLn('Enter path to input file: ');

ReadLn(Reader);

AssignFile(InF, Reader);

Reset(InF);

Valid := true;

except

WriteLn('Invalid path');

end;

until Valid;

ReadLn(InF, Reader);

if Length(Reader) < 256 then

begin

WriteLn('Entered String: ', Reader);

for I := 1 to Length(Reader) do

include(S, Reader[I]);

end

else

WriteLn('Entered set is too large');

Close(InF);

end;

procedure OutputData(Answer: SetOfChar);

var

OutF: TextFile;

I: Integer;

Reader: String;

begin

WriteLn('Enter path to output file: ');

ReadLn(Reader);

AssignFile(OutF, Reader);

Rewrite(OutF);

for I := 0 to 255 do

if Chr(I) in Answer then

begin

Write(OutF, Chr(I));

Write(Chr(I));

end;

WriteLn(OutF, ' - answer.');

WriteLn(' - answer.');

Close(OutF);

end;

procedure Main();

var

S: SetOfChar;

Reader: Char;

Valid: Boolean;

begin

WriteLn('This program filters out everything except numbers

and arithmetic operations, and create new string.'

+ #13#10);

S := [];

Valid := false;

repeat

WriteLn('Enter "C" if you want to input data from

console or' + #13#10 + '"F" if you want to input

data from file');

ReadLn(Reader);

case Reader of

'C':

begin

InputData(S);

Valid := true;

end;

'F':

begin

InputDataFromFile(S);

Valid := true;

end;

end;

until Valid;

if SizeOf(S) > 0 then

OutputData(Intersection(S))

else

WriteLn('Set is empty');

ReadLn;

end;

begin

Main();

end.

**Код программы**

**(C)**

#include <stdio.h>

#include <string.h>

void inputData(char \*s) {

printf("Enter string: \n");

fflush(stdin);

fgets(s, 100, stdin); //42

}

void inputDataFromFile(char \*s) {

FILE \*inF = fopen("D:\\University\\OAiP\\LAB3\\Goal

2\\C\\goal\\input.txt", "r");

fgets(s, 100, inF);

printf("Entered string: %s\n", s);

fclose(inF);

}

void creating(char \*s, char \*answer) {

char pattern[18] = {'0', '1', '2', '3', '4', '5', '6', '7',

'8', '9', '.', '^', '!', '+', '-', '\*', '/'};

int j = 0;

for(int i = 0; i < strlen(s); i++) {

if (strchr(pattern, s[i])) {

answer[j] = s[i];

j++;

}

}

answer[j] = '\0';

}

void outputData(char \*answer) {

FILE \*outF = fopen("D:\\University\\OAiP\\LAB3\\Goal

2\\C\\goal\\output.txt", "w");

fprintf(outF, "%s - answer", answer);

printf("%s - answer", answer);

fclose(outF);

}

void main() {

char s[101], answer[101], choice;

int valid = 1;

printf("This program filters out everything except numbers

and arithmetic operations, and create new string.

\n\n");

do {

printf("Enter \"C\" if you want to input data from

console or \n\"F\" if you want to input data from

file: \n");

fflush(stdin);

scanf("%c", &choice);

switch (choice) {

case 'C' :

inputData(s);

valid = 0;

break;

case 'F' :

inputDataFromFile(s);

valid = 0;

break;

}

} while (valid);

s[strlen(s) - 1] = '\0';

creating(s, answer);

outputData(answer);

}

**Код программы**

**(Java)**

import java.util.Scanner;

import java.io.\*;

public class Main {

public static void main(String[] args) throws IOException {

Scanner in = new Scanner(System.in);

String s = "";

boolean valid = true;

System.out.println("This program filters out everything

except numbers and arithmetic operations, and

create new string. \n");

do {

System.out.println("Enter \"C\" if you want to input

data from console or \n\"F\" if you want to

input data from file");

switch (in.nextLine()) {

case "C" :

s = inputData();

valid = false;

break;

case "F" :

s = inputDataFromFile();

valid = false;

break;

}

} while (valid);

outputData(creating(s));

}

public static String inputData() {

Scanner in = new Scanner(System.in);

System.out.println("Enter string: ");

return in.nextLine();

}

public static String inputDataFromFile() throws IOException{

Scanner in = new Scanner(System.in);

Scanner fileScanner = in;

String s = "";

boolean valid = true;

FileReader inF = new FileReader("input.txt");

do {

try {

System.out.println("Enter path to input file:

");

inF = new FileReader(in.nextLine());

fileScanner = new Scanner(inF);

valid = false;

} catch (Exception e) {

System.out.println("INPUT ERROR");

}

} while (valid);

s = fileScanner.nextLine();

inF.close();

System.out.println("Entered string: " + s);

return s;

}

public static String creating(String s) {

String answer = "";

for(int i = 0; i < s.length(); i++) {

if (("" + s.charAt(i)).matches("[\\d.^!+\\-\*/]\*")) {

answer += s.charAt(i);

}

}

return answer;

}

public static void outputData(String answer) throws

IOException {

Scanner in = new Scanner(System.in);

System.out.println("Enter path to output file: ");

FileWriter outF = new FileWriter(in.nextLine());

outF.write(answer + " - answer");

System.out.println(answer + " - answer");

outF.close();

}

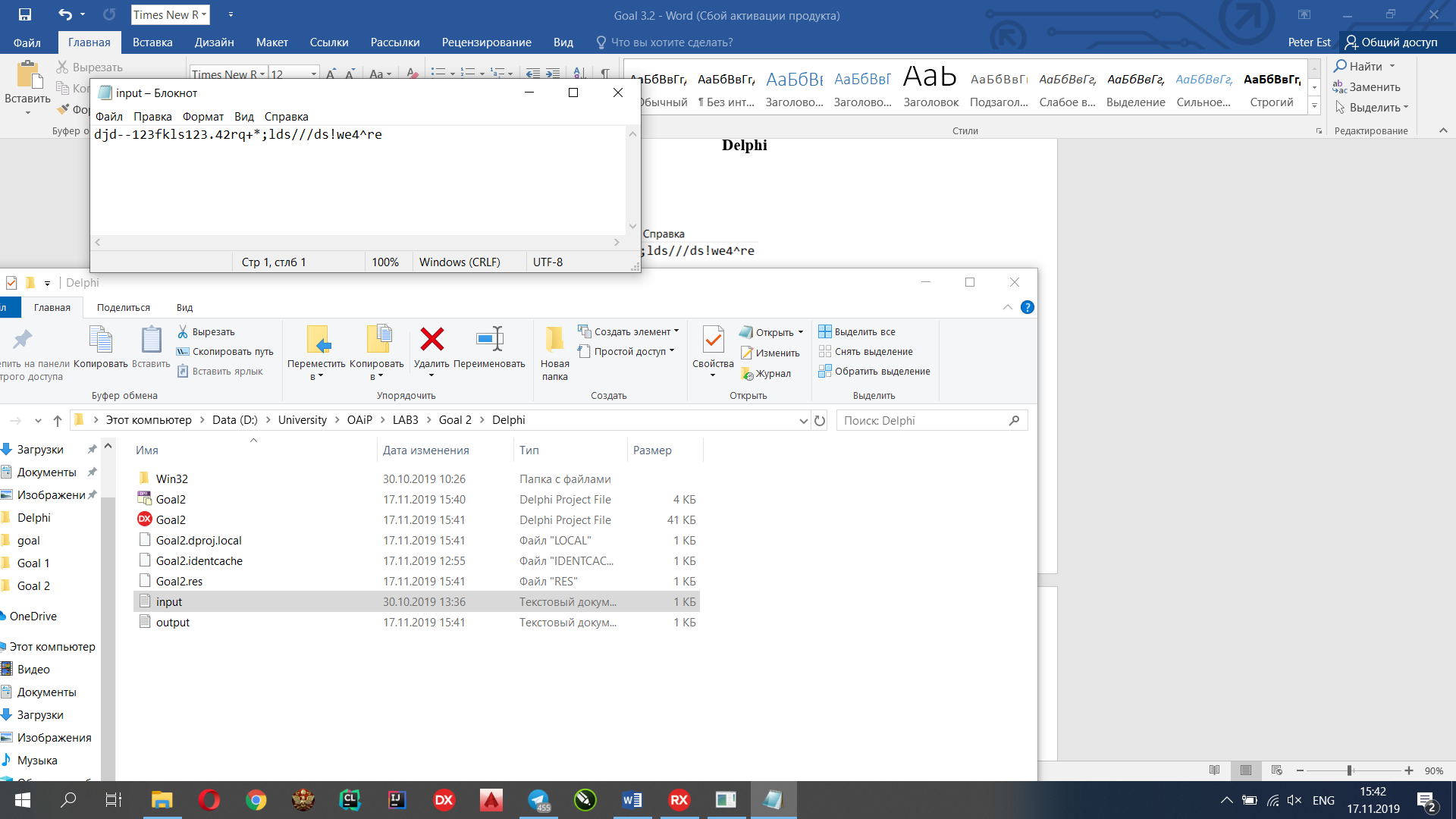
}

**Блок-схема**

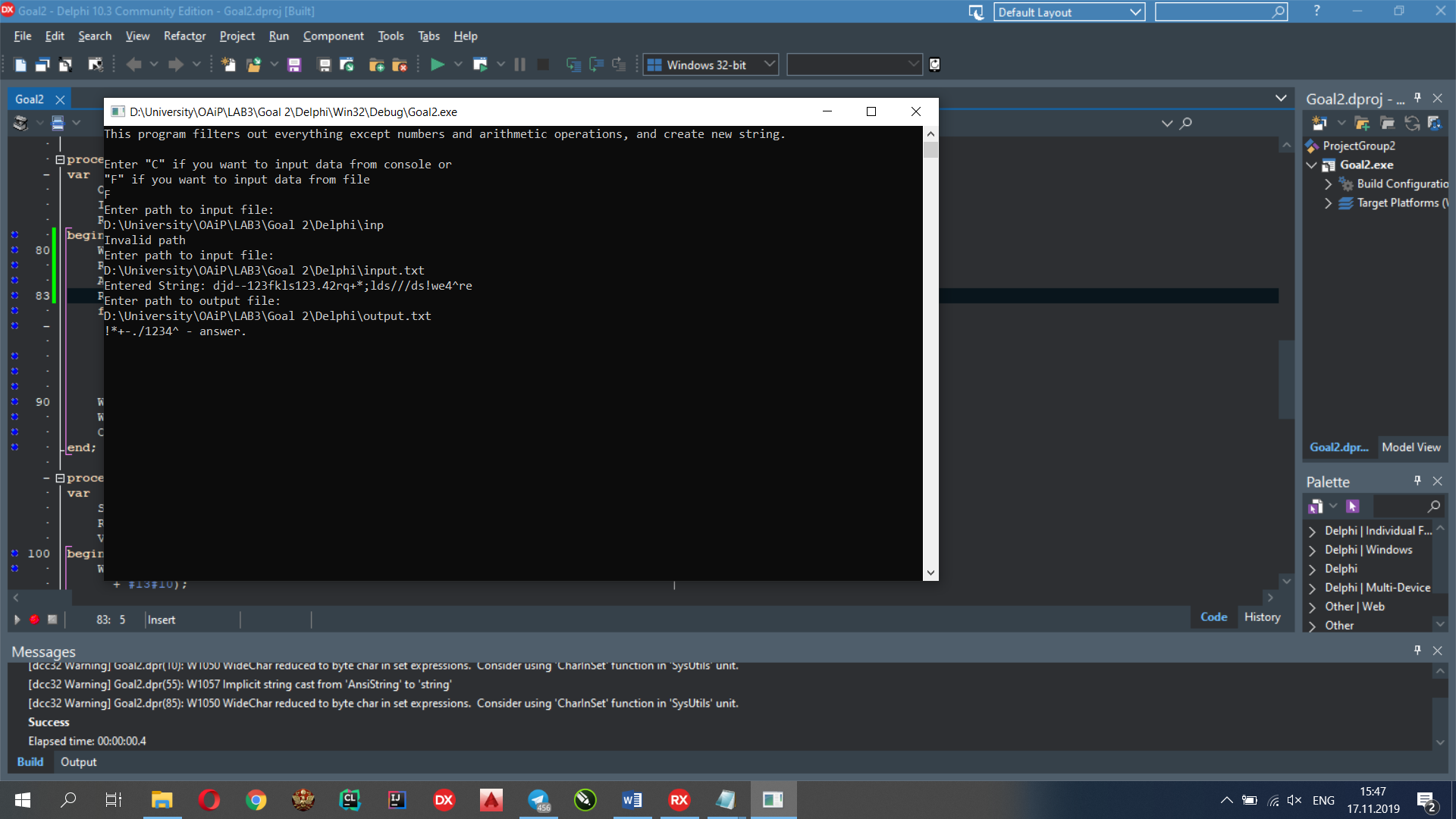
**Работа программы**

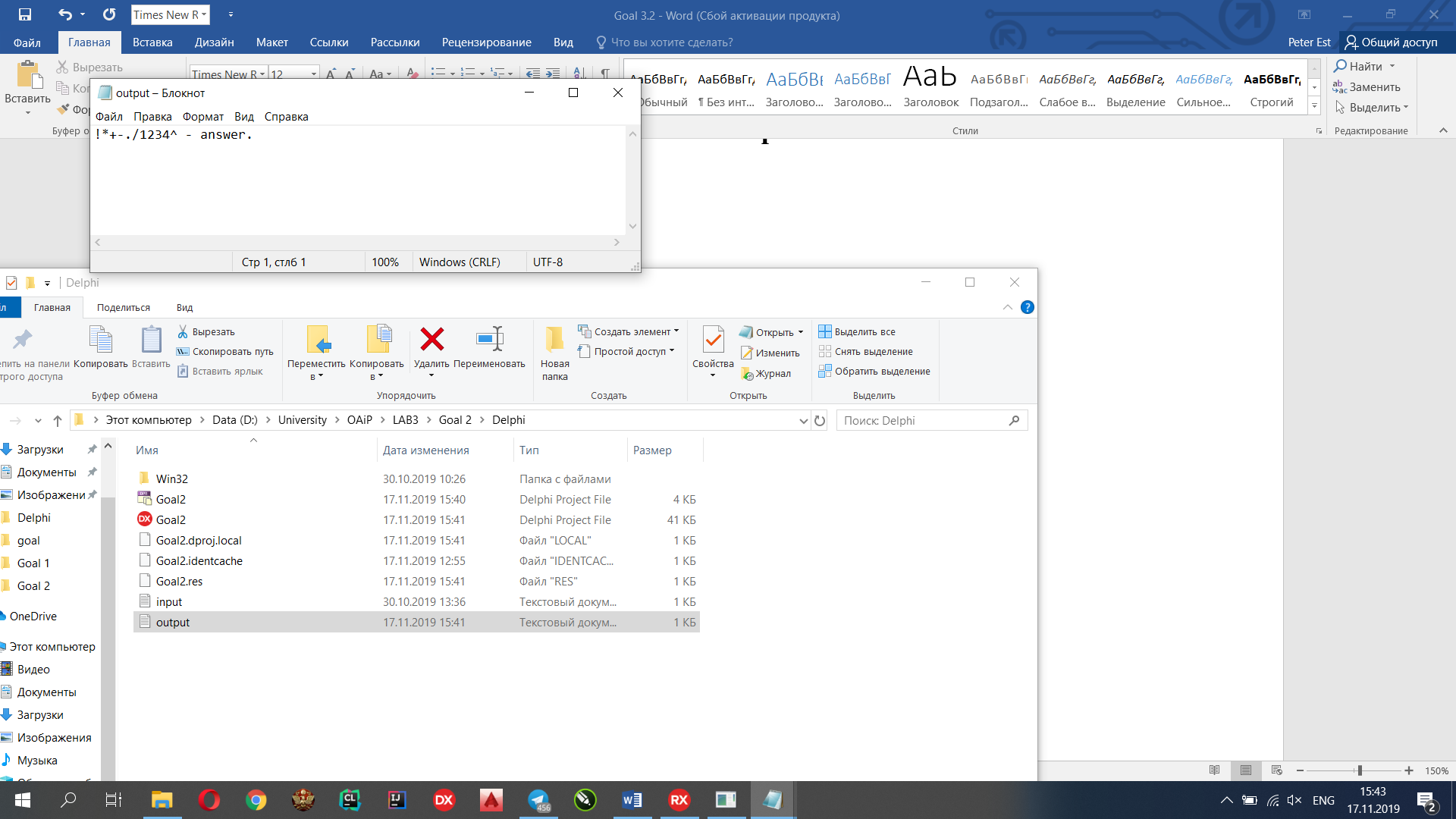
**Delphi**

Input:

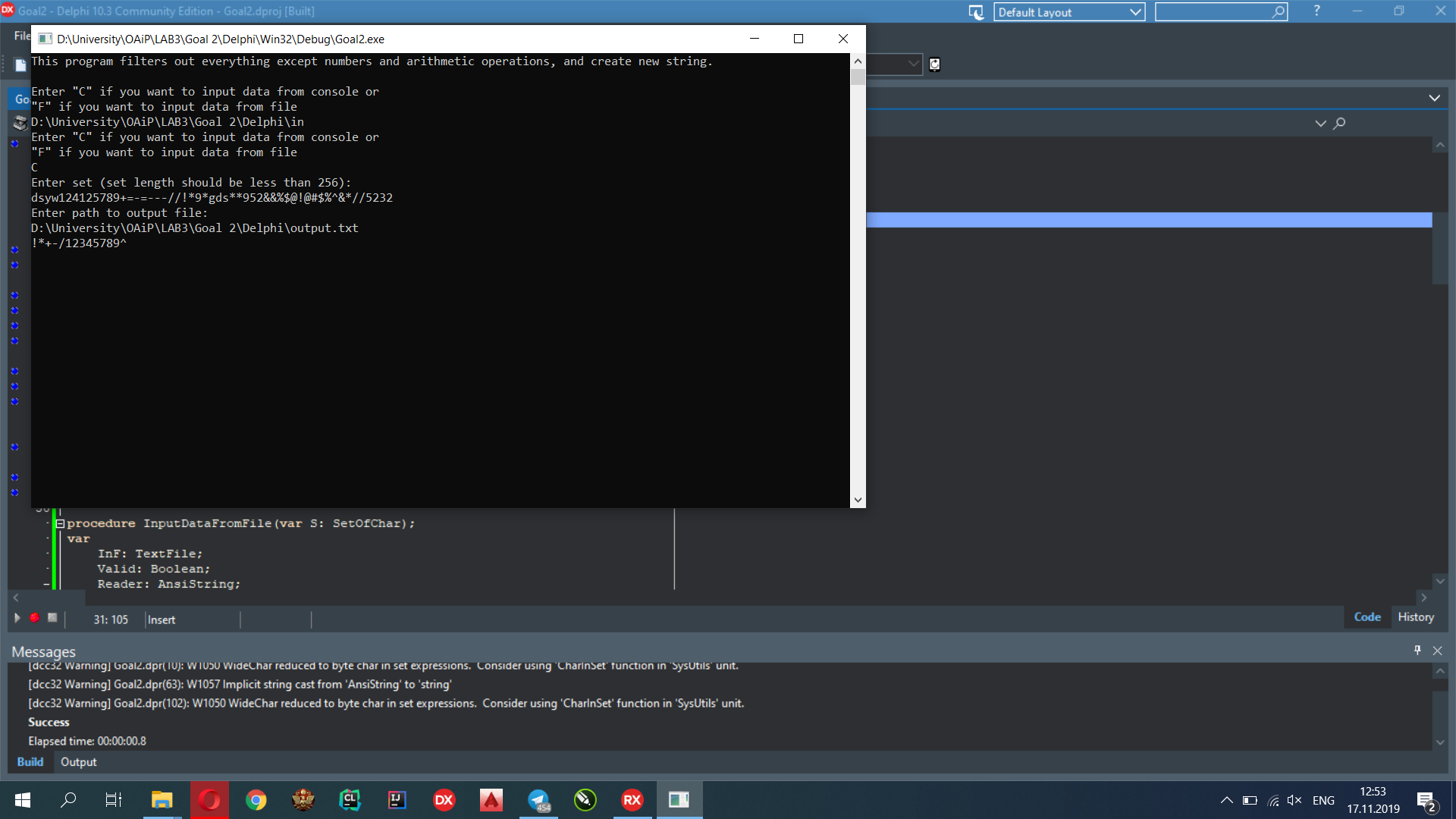


Output:



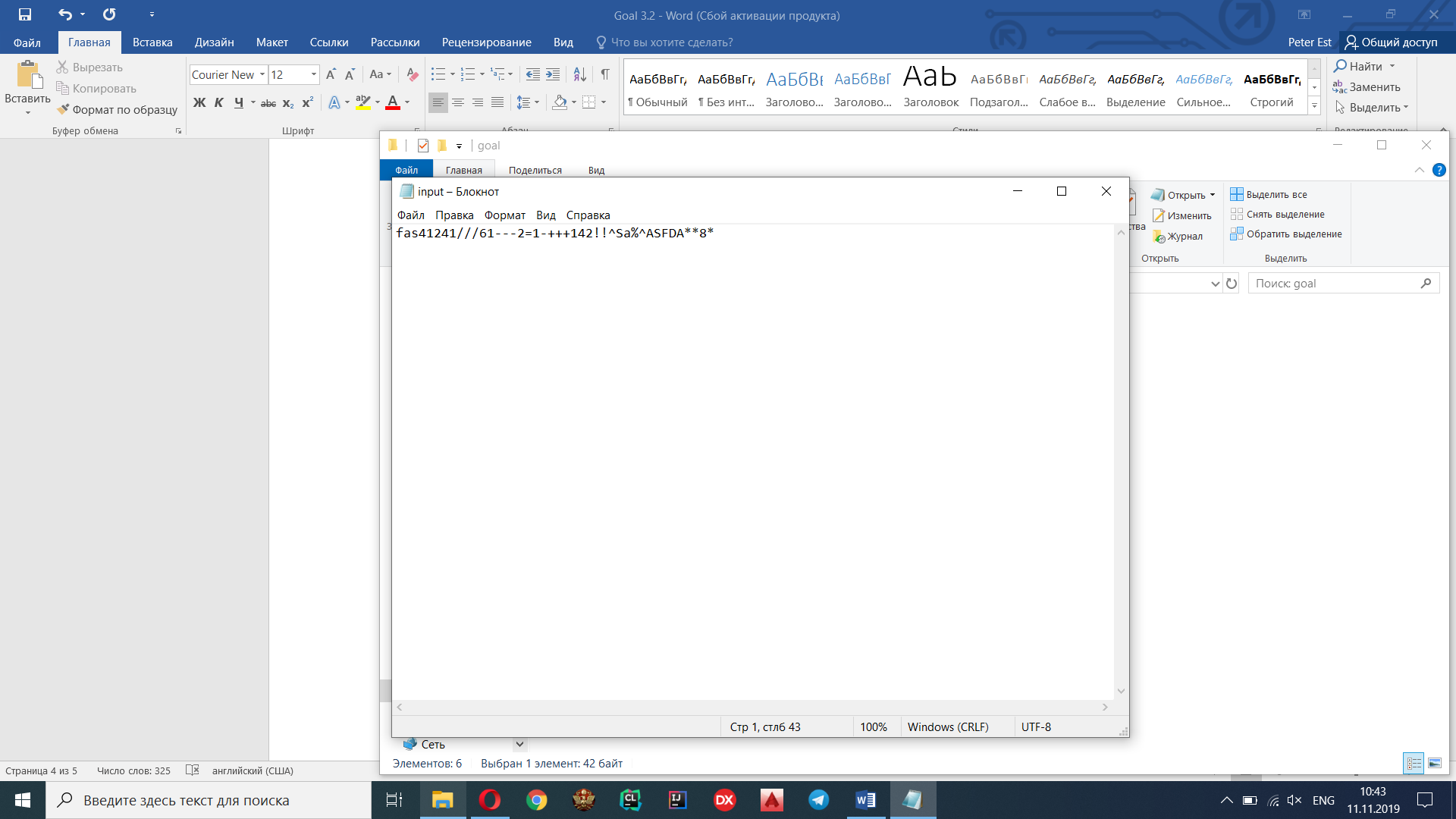


Console:

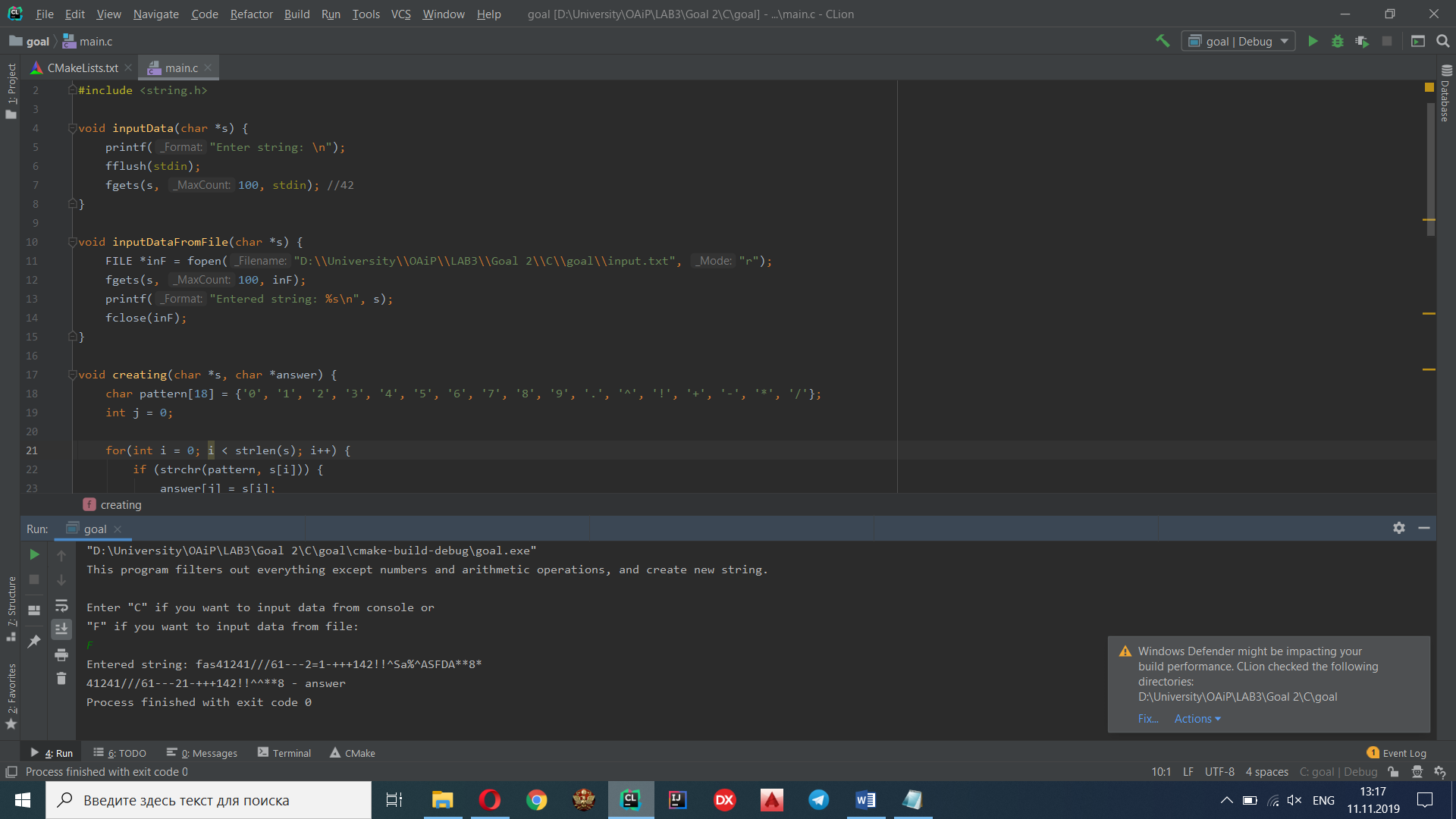


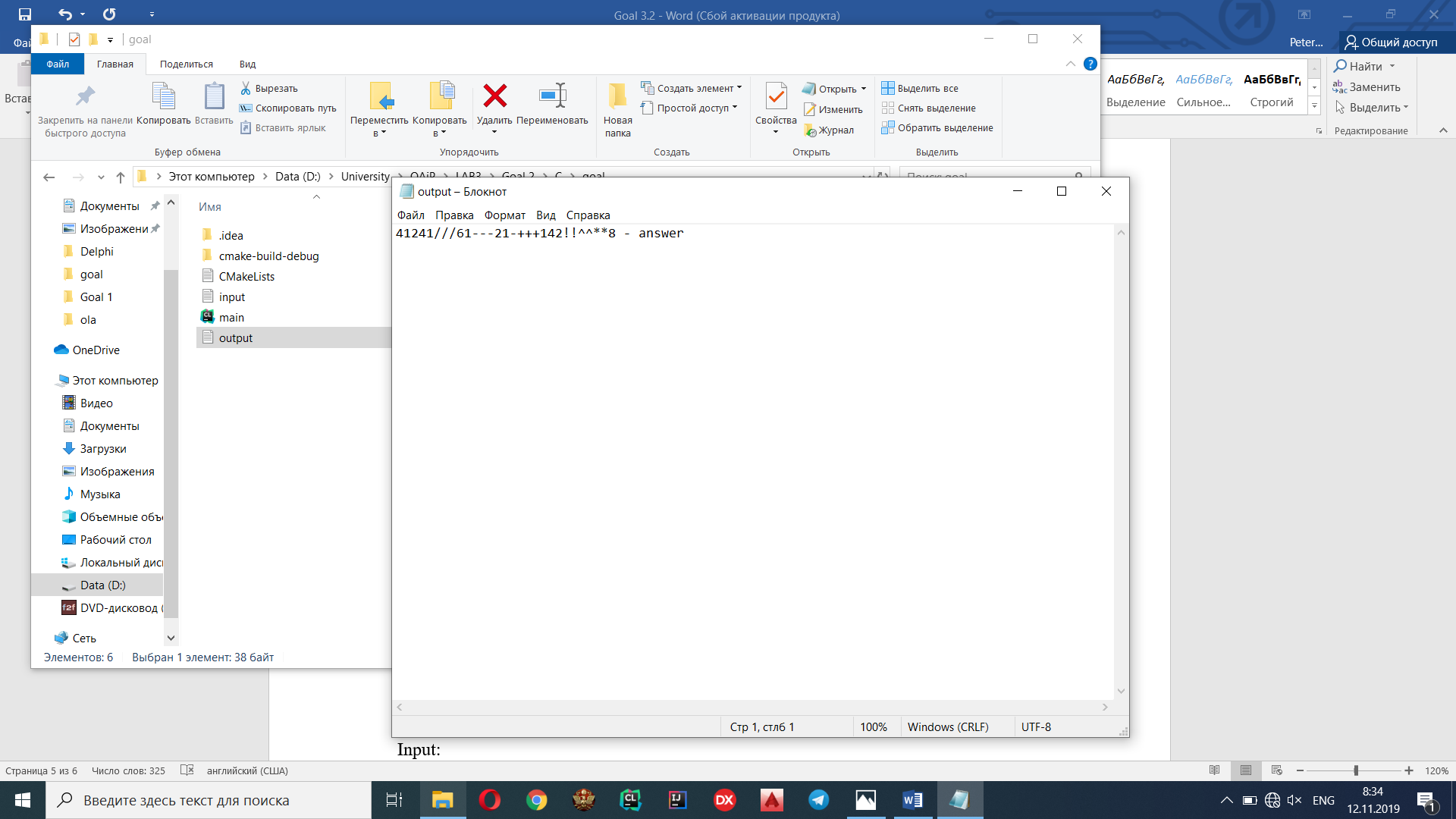
**C**

Input:

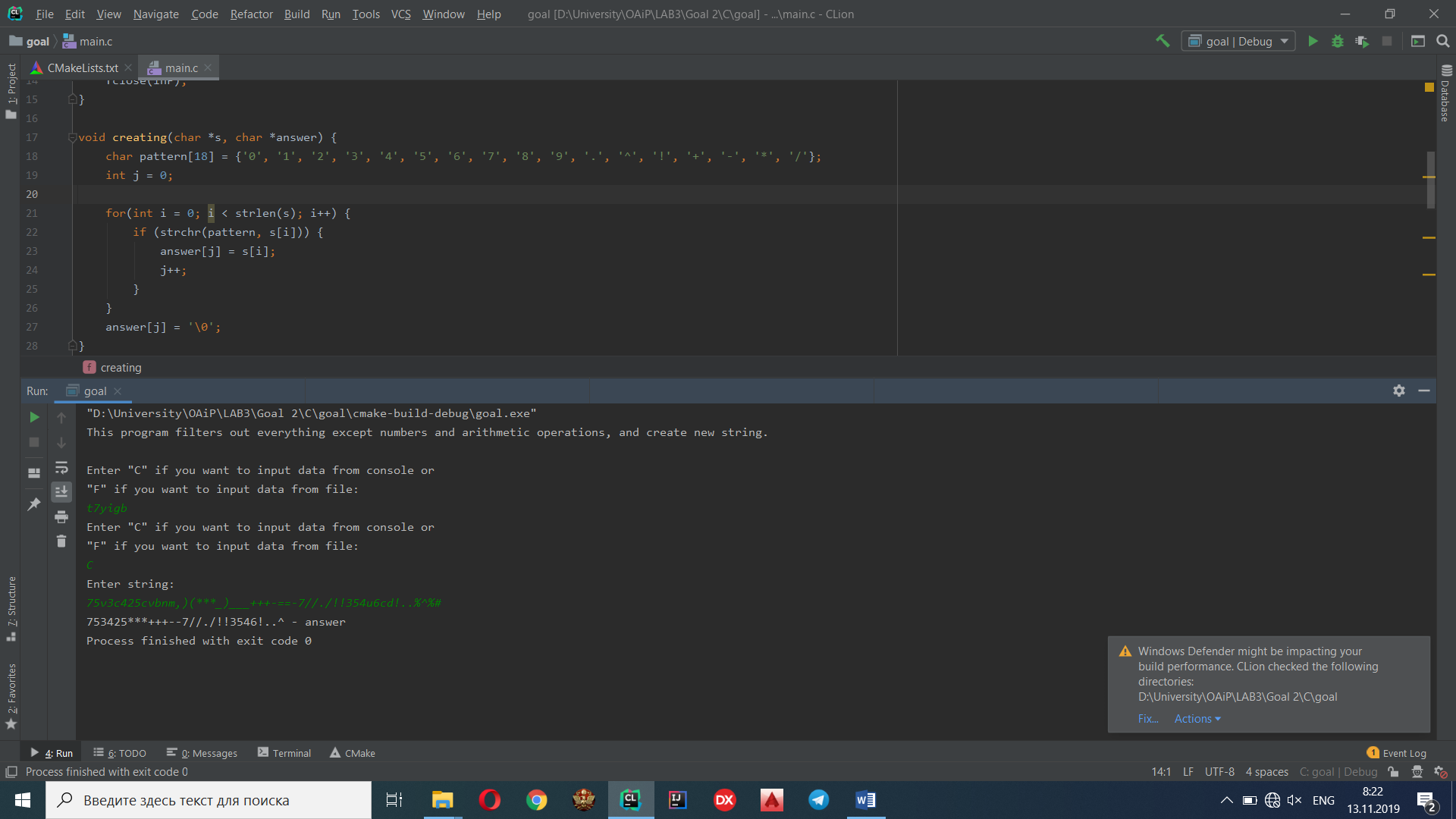


Output:



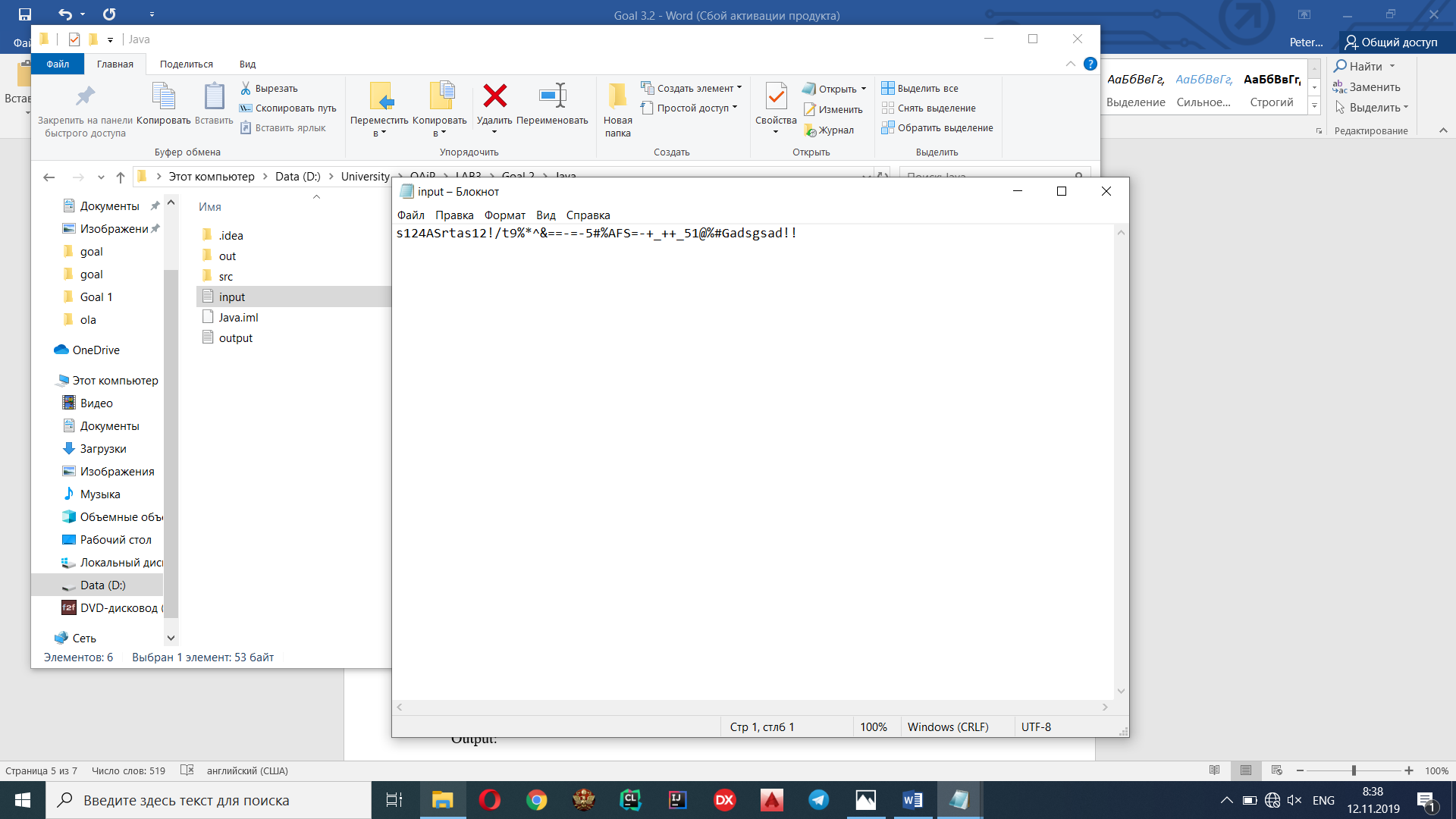


Console:

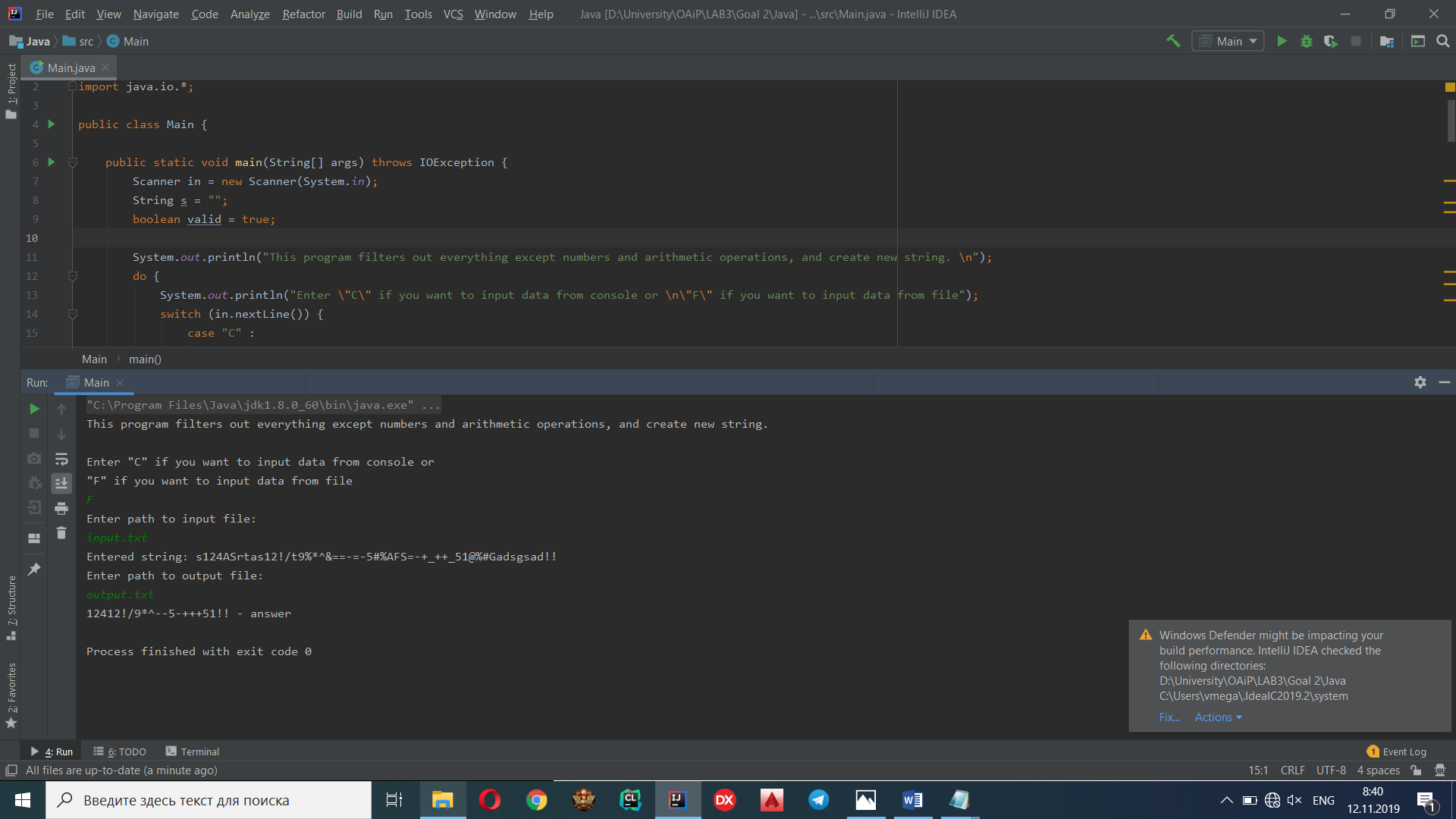


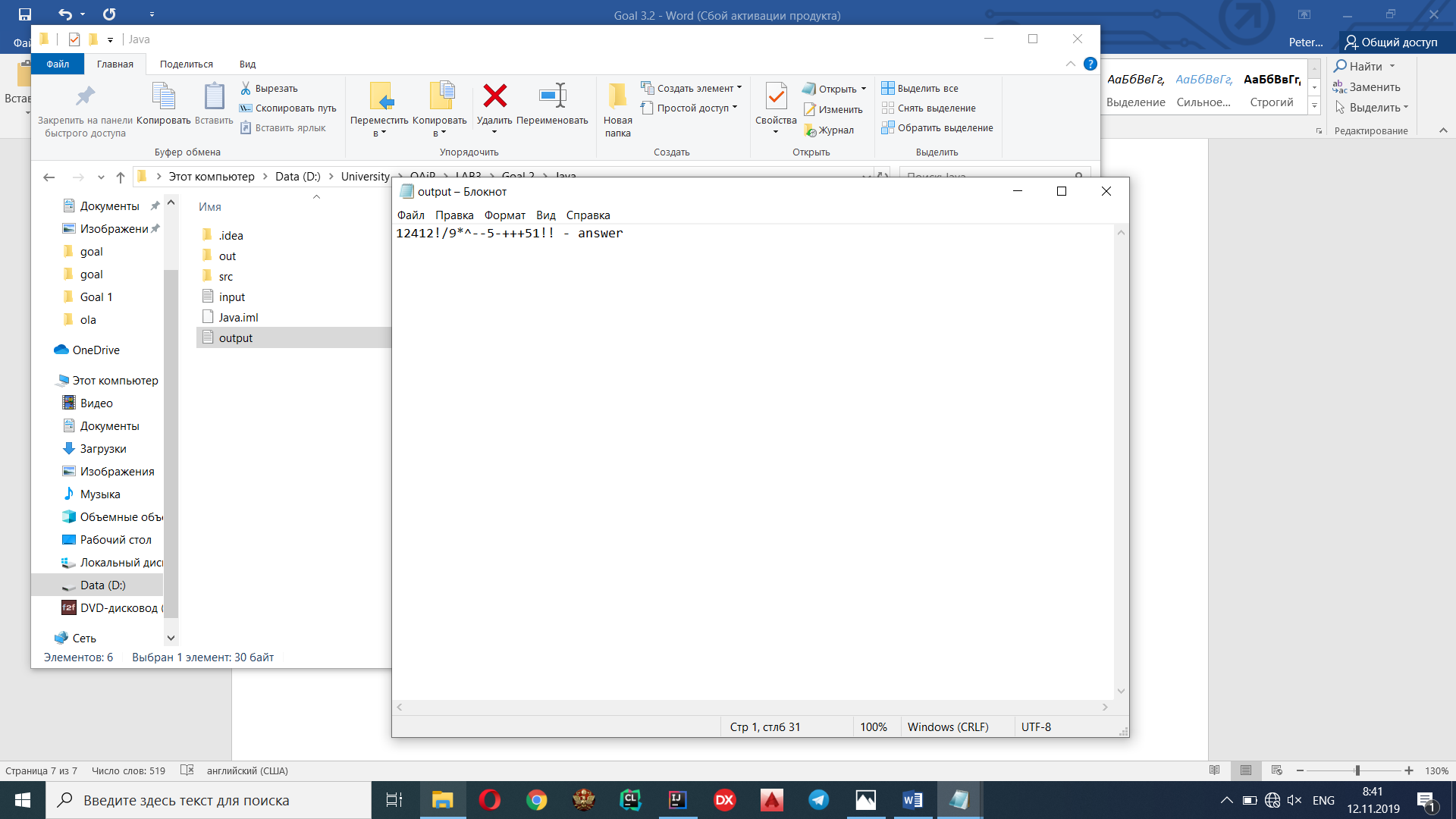
**Java**

Input:



Output:





Console:

