УО «Белорусский государственный университет информатики и

радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе № 2.4

по предмету

Основы Алгоритмизации и Программирования

Вариант 4

Выполнил

Воривода М.А.

Проверила

Данилова Г.В.

Группа:

951007

Минск 2019

**Задание**

Даны два упорядоченных по возрастанию массива целых чисел: A, состоящего из n элементов, и B, состоящего из m элементов. Выполнить слияние этих массивов в один упорядоченный массив C.

**Код программы**

**(Delphi)**

program Goal4;

uses

SysUtils;

type

ArrayOfInteger = array of Integer;

function sort(arr: ArrayOfInteger): ArrayOfInteger;

var

i, j, Value: Integer;

begin

for i := 0 to Length(arr) - 1 do

for j := 0 to Length(arr) - 1 do

if arr[i] < arr[j] then

begin

Value := arr[j];

arr[j] := arr[i];

arr[i] := Value;

end;

Result := arr;

end;

function splitToInt(S: String; Size: Integer): ArrayOfInteger;

var

arr: ArrayOfInteger;

i: Integer;

begin

SetLength(arr, Size);

i := 0;

while (pos(', ', S) > 0) and (i < Size) do

begin

arr[i] := StrToInt(Copy(S, 1, (pos(', ', S) - 1)));

delete(S, 1, pos(', ', S) + 1);

inc(i);

end;

arr[i] := StrToInt(S);

Result := arr;

end;

function arrConcat(A, B: ArrayOfInteger): ArrayOfInteger;

var

C: ArrayOfInteger;

i: Integer;

begin

SetLength(C, Length(A) + Length(B));

for i := 0 to Length(A) - 1 do

C[i] := A[i];

for i := Length(A) to Length(B) + Length(A) - 1 do

C[i] := B[i - Length(A)];

Result := C;

end;

procedure arrOut(arr: ArrayOfInteger; var outF: TextFile);

var

i: Integer;

str: String;

begin

str := IntToStr(arr[0]);

for i := 1 to Length(arr) - 1 do

str := str + ', ' + IntToStr(arr[i]);

Writeln(outF, str);

Writeln(#13#10 + str);

end;

procedure main();

var

A, B, C: ArrayOfInteger;

M, N: Integer;

Reader: String;

inF, outF: TextFile;

Valid: boolean;

begin

Valid := false;

repeat

try

Writeln('Enter input file directory: ');

ReadLn(Reader);

AssignFile(inF, Reader);

Reset(inF);

Valid := true;

except

Writeln('File not found');

end;

until Valid;

Writeln('Enter output file directory: ');

ReadLn(Reader);

AssignFile(outF, Reader);

Rewrite(outF);

try

ReadLn(inF, Reader);

N := StrToInt(Reader);

ReadLn(inF, Reader);

M := StrToInt(Reader);

ReadLn(inF, Reader);

A := splitToInt(Reader, N);

ReadLn(inF, Reader);

B := splitToInt(Reader, M);

except

Writeln(outF, 'INPUT ERROR');

Writeln('INPUT ERROR');

Valid := false;

end;

if Valid then

begin

SetLength(A, N);

SetLength(B, M);

SetLength(C, M + N);

A := sort(A);

B := sort(B);

C := sort(arrConcat(A, B));

arrOut(C, outF);

Close(inF);

Close(outF);

end;

ReadLn;

end;

begin

main();

end.

**Код программы**

**(C)**

#include <stdio.h>

void sort(int \*arr, int Size) {

int Value, i, j;

for (i = 0; i < Size; i++) {

for (j = 0; j < Size; j++) {

if (arr[i] < arr[j]) {

Value = arr[j];

arr[j] = arr[i];

arr[i] = Value;

}

}

}

}

void splitToInt(int \*arr, int Size, FILE \*inF) {

int i;

for (i = 0; i < Size; i++) {

fscanf(inF, "%d", &arr[i]);

}

}

int arrConcat(int \*C, int \*A, int SizeA, int \*B, int SizeB) {

int i;

for (i = 0; i < SizeA; i++) {

C[i] = A[i];

}

for (i = SizeA; i - SizeA < SizeB; i++) {

C[i] = B[i - SizeA];

}

}

void arrOut(int \*arr, int Size, FILE \*outF) {

int i;

for (i = 0; i < Size; i++) {

fprintf(outF, "%d ", arr[i]);

printf("%d ", arr[i]);

}

}

void main() {

FILE \*inF = fopen("D:\\University\\OAiP\\LAB2\\C\\Goal

4\\in.txt", "r");

FILE \*outF = fopen("D:\\University\\OAiP\\LAB2\\C\\Goal

4\\out.txt", "w");

int N, M;

fscanf(inF, "%d\n%d", &N, &M);

int A[N];

int B[M];

int SizeC = M + N;

int C[SizeC];

splitToInt(A, N, inF);

splitToInt(B, M, inF);

sort(A, N);

sort(B, M);

arrConcat(C, A, N, B, M);

sort(C, sizeof(C)/4);

arrOut(C, N + M, outF);

fclose(inF);

fclose(outF);

}

**Код программы**

**(Java)**

import java.util.Scanner;

import java.io.\*;

public class Main {

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

boolean inValid = true;

int N = 0, M = 0;

int[] A = new int[0], B = new int[0];

Scanner in = new Scanner(System.in);

do {

try {

System.out.println("Enter input file directory:

");

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

inValid = false;

} catch (Exception e) {

System.out.println("File not found");

}

} while (inValid);

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

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

Scanner fileScanner = new Scanner(inF);

try {

N = Integer.parseInt(fileScanner.nextLine());

M = Integer.parseInt(fileScanner.nextLine());

A = new int[N];

B = new int[M];

} catch (Exception e) {

inValid = false;

}

try {

A = splitToInt(fileScanner.nextLine(), N);

B = splitToInt(fileScanner.nextLine(), M);

} catch (Exception e) {

inValid = false;

}

if (inValid) {

outF.write("INPUT ERROR");

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

} else {

A = sort(A);

B = sort(B);

arrOut(sort(arrConcat(A, B)), outF);

}

inF.close();

outF.close();

}

public static int[] sort(int[] arr) throws Exception {

int Value, i, j;

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

for (j = 0; j < arr.length; j++) {

if (arr[i] < arr[j]) {

Value = arr[j];

arr[j] = arr[i];

arr[i] = Value;

}

}

}

return arr;

}

public static int[] splitToInt(String str, int Size) throws

Exception {

int[] A = new int[Size];

int i = 0;

while (str.indexOf(", ") > 0 && i < Size) {

A[i] = Integer.parseInt(str.substring(0,

str.indexOf(", ")));

str = str.replace(A[i] + ", ", "");

i++;

}

A[i] = Integer.parseInt(str);

return A;

}

public static int[] arrConcat(int[] A, int[] B) throws

Exception {

int i;

int[] C = new int[A.length + B.length];

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

C[i] = A[i];

}

for (i = A.length; i - A.length < B.length; i++) {

C[i] = B[i - A.length];

}

return C;

}

public static void arrOut(int[] arr, FileWriter outF) throws

Exception {

int i;

String str;

str = "[" + arr[0];

for (i = 1; i < arr.length; i++) {

str = str + ", " + arr[i];

}

str += "]";

outF.write(str);

System.out.println(str);

}

}

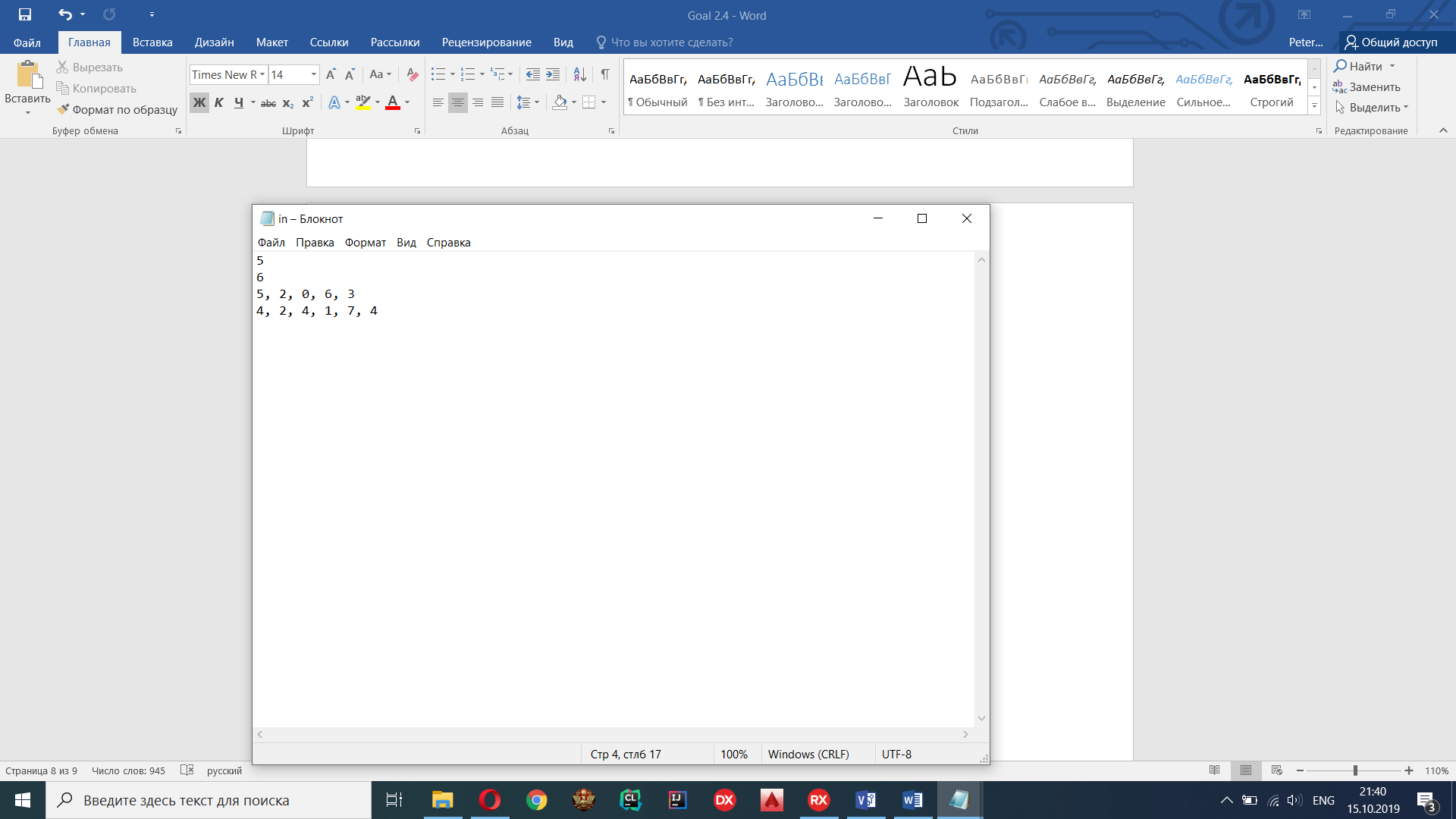
**Блок-схема**



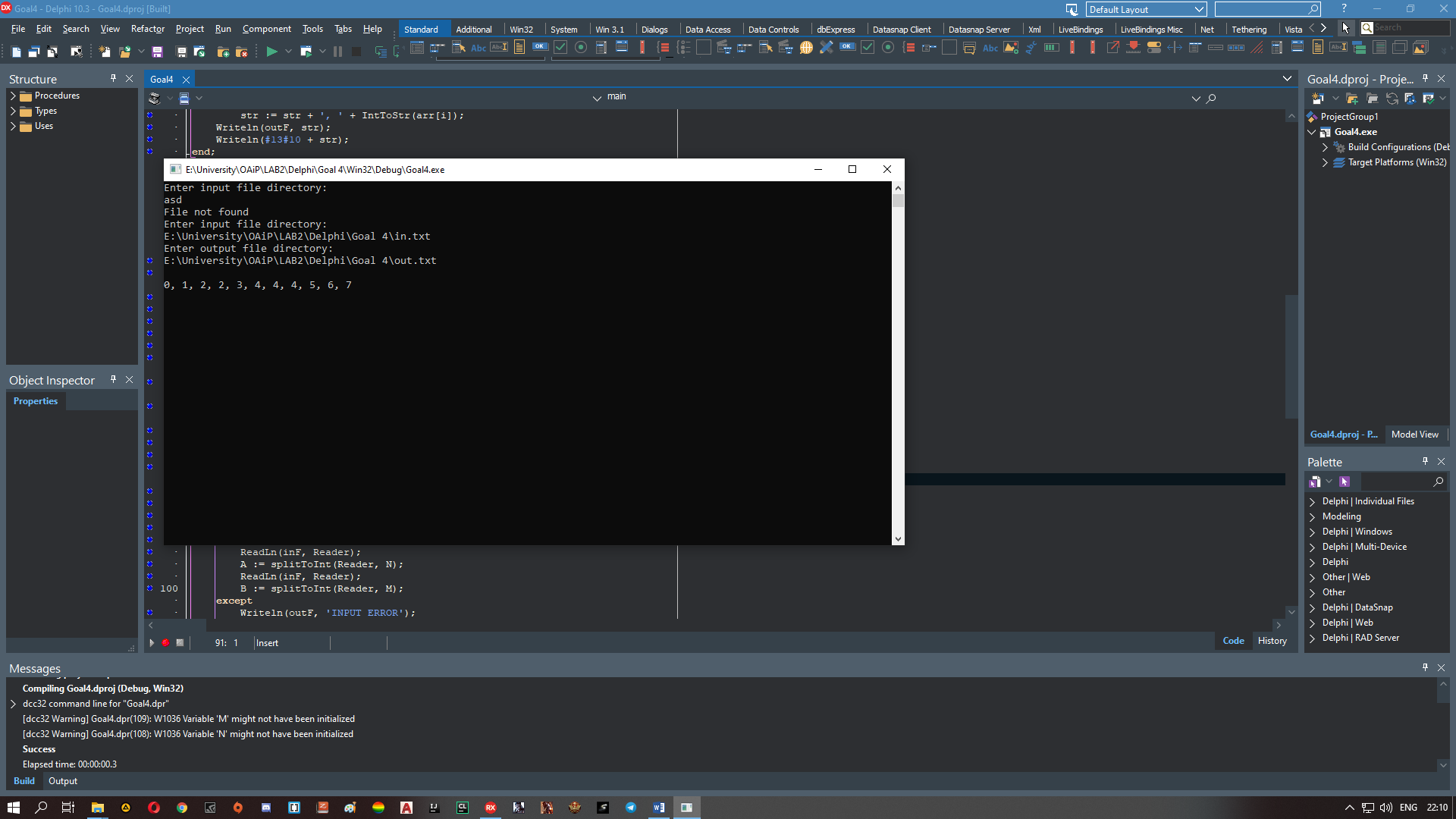
**Работа программы**

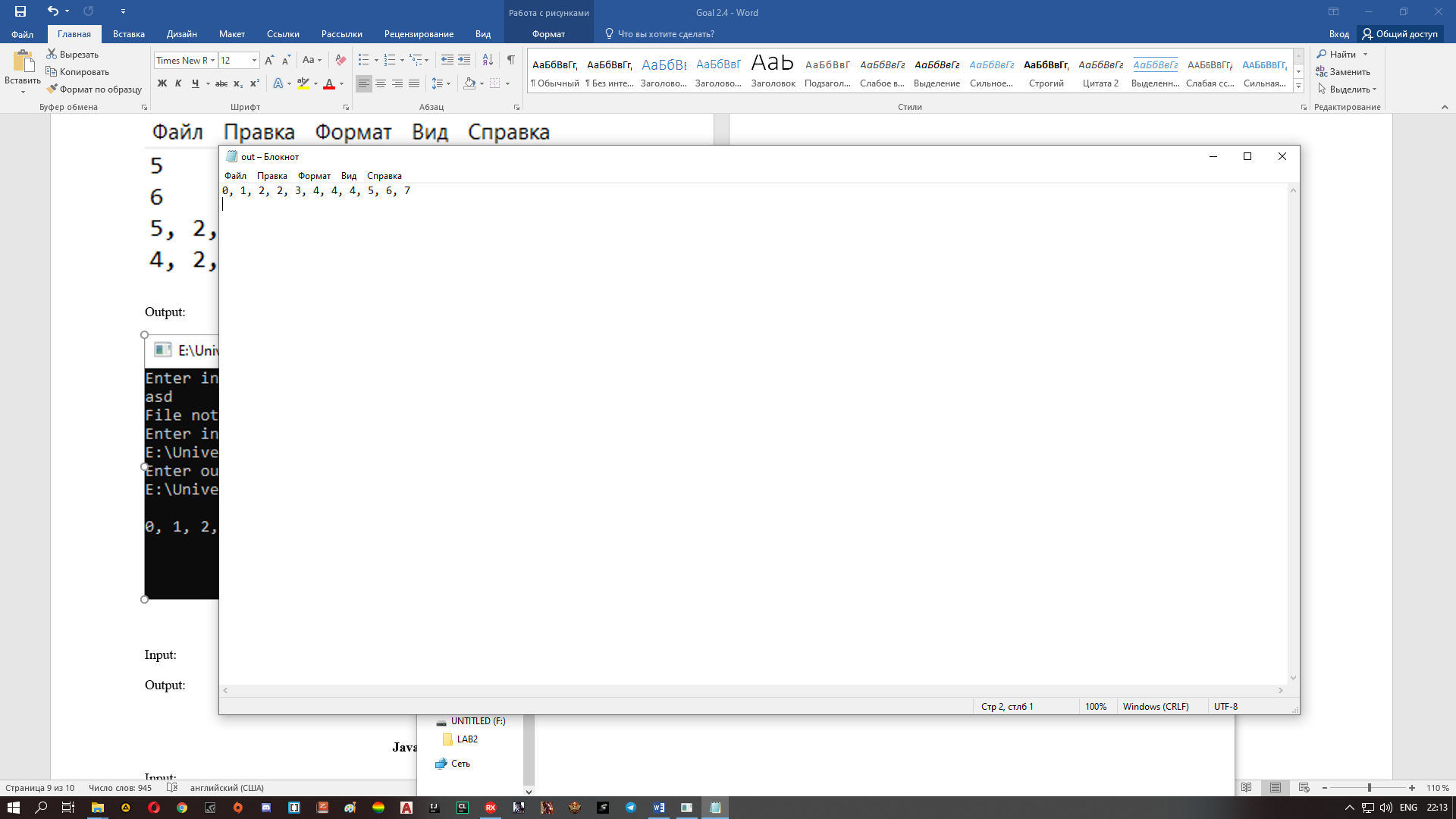
**Delphi**

Input:



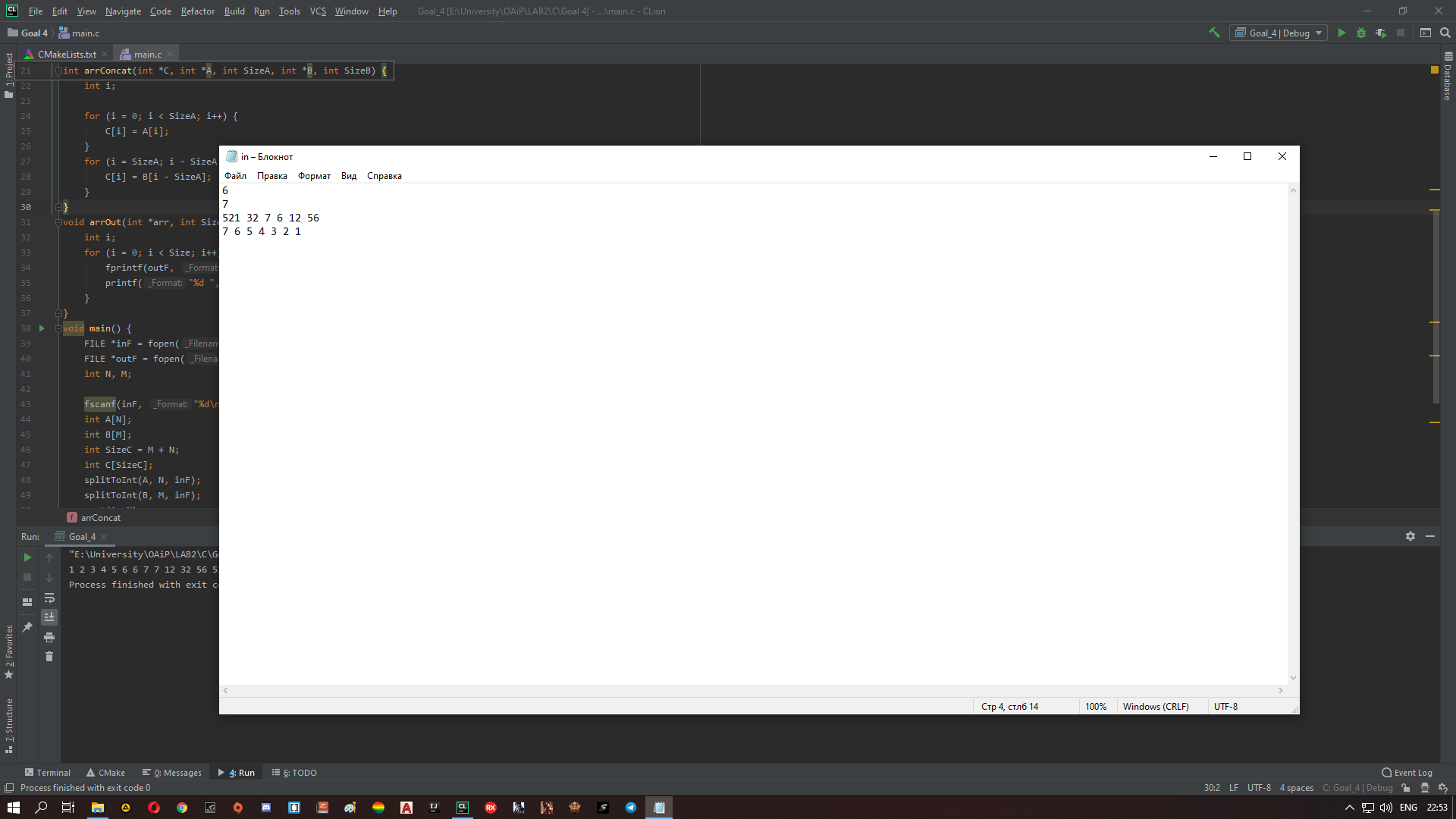
Output:



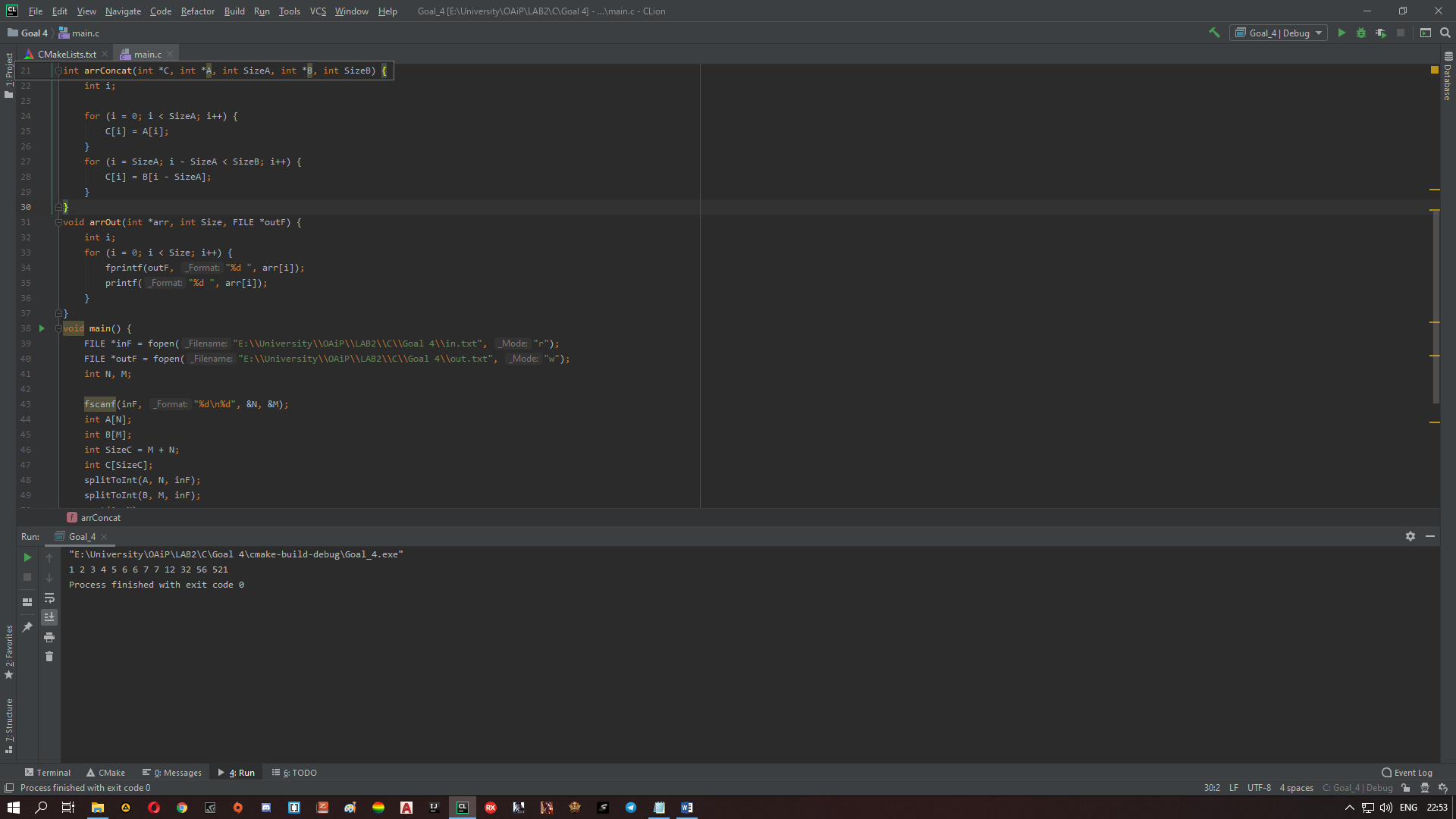


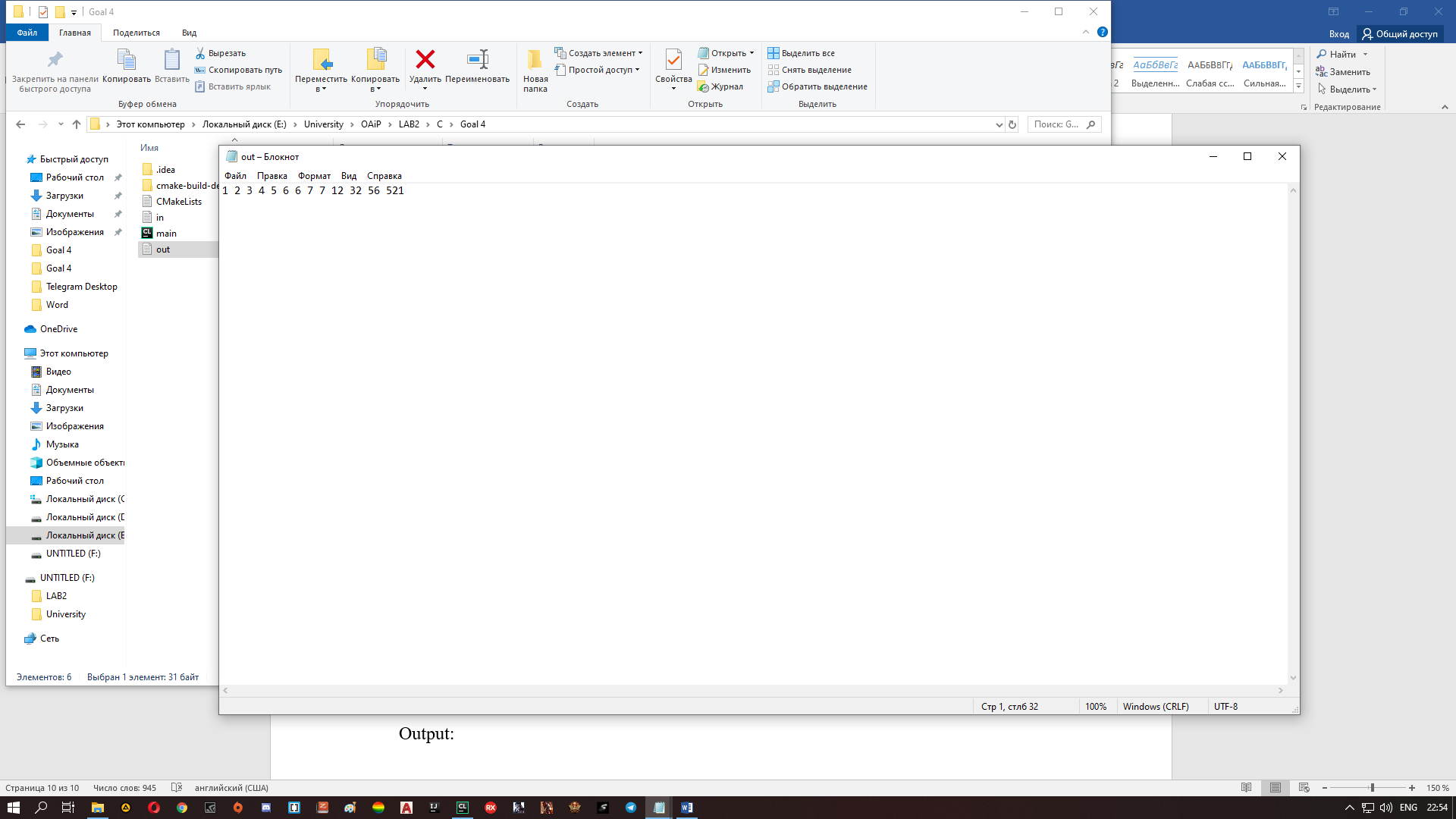
**C**

Input:



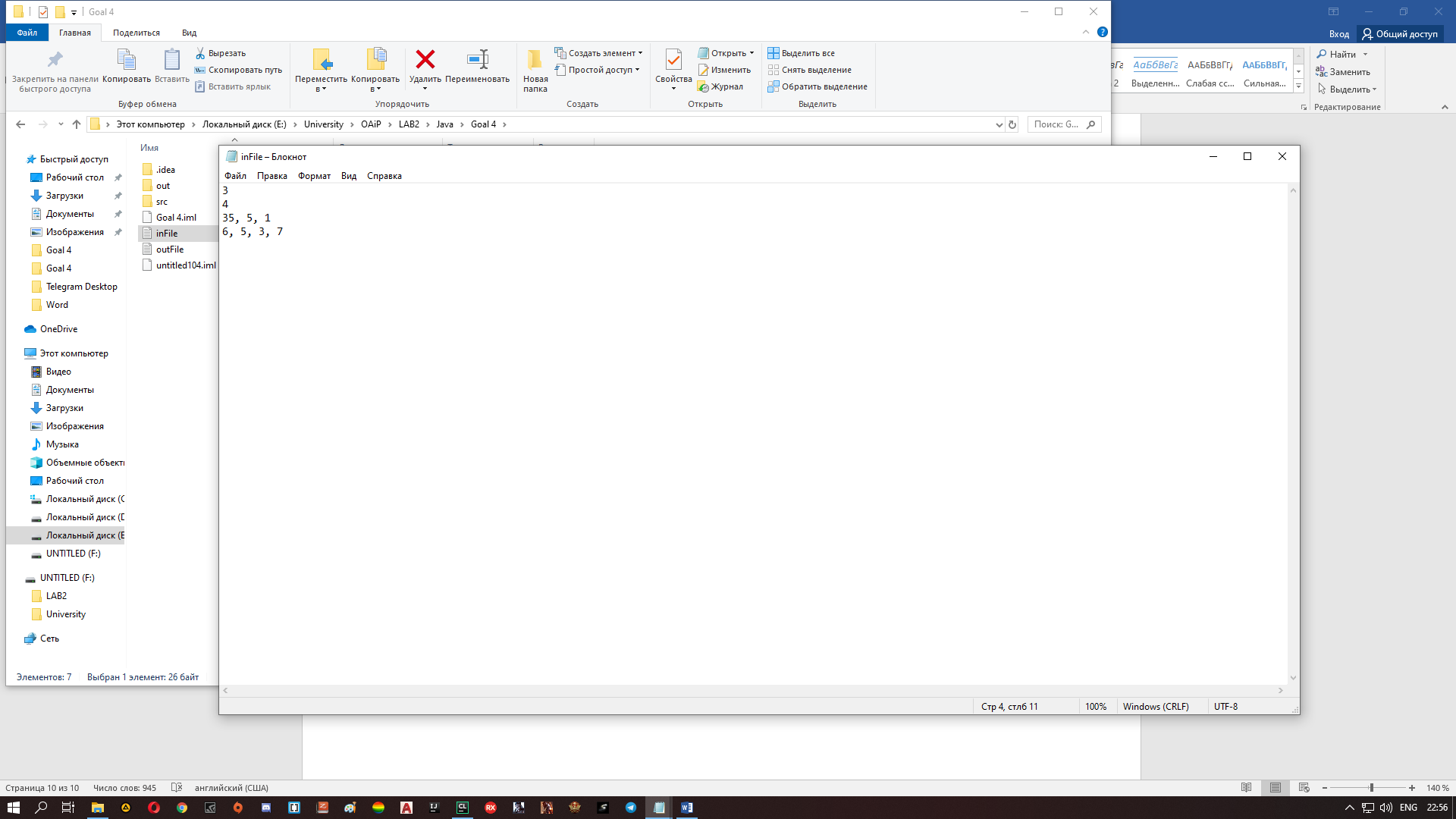
Output:





**Java**

Input:



Output:

