27.11.2020

**Algorytmy i struktury danych**

Projekt nr 1

Katarzyna Jadowska

154319

FS0-DI

gr. P3

# Wstęp

Celem projektu było stworzenie programu wyznaczającego maksymalny iloczyn dwóch elementów z tablicy liczb całkowitych. Do głównych cech programu należy odczyt danych wejściowych oraz zapis wyników do pliku tekstowego.

# Opis problemu

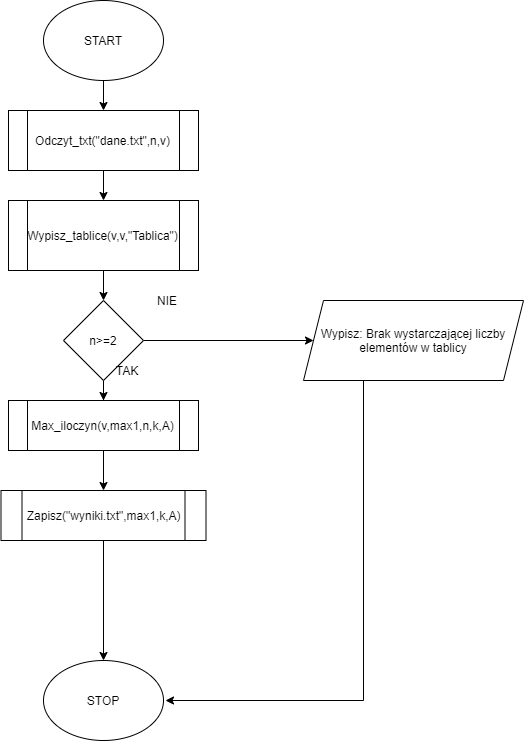
Wartość największego iloczynu dwóch liczb w tablicy uzyskana może być dla dwóch największych lub dwóch najmniejszych liczb. Prostym rozwiązaniem tego problemu jest wymnożenie przez siebie wszystkich elementów tablicy i sprawdzenie, który iloczyn jest największy. Innym rozwiązaniem może być posortowanie tablicy, a następnie sprawdzenie iloczynu dwóch pierwszych oraz dwóch ostatnich elementów. Szczególny przypadek stanowi sytuacja, w której największy iloczyn jest równy 0. W takim przypadku jeśli tablica zawiera więcej niż trzy elementy należy pamiętać, że maksymalny iloczyn daje również para liczb [0,0].

W projekcie opracowano algorytm, który oblicza maksymalny iloczyn najpierw dla liczb dodatnich a następnie szuka maksymalnego iloczynu wśród liczb ujemnych. Wartość maksymalnego iloczynu wraz z liczbami, które go wyznaczają zapisywane są do pliku.

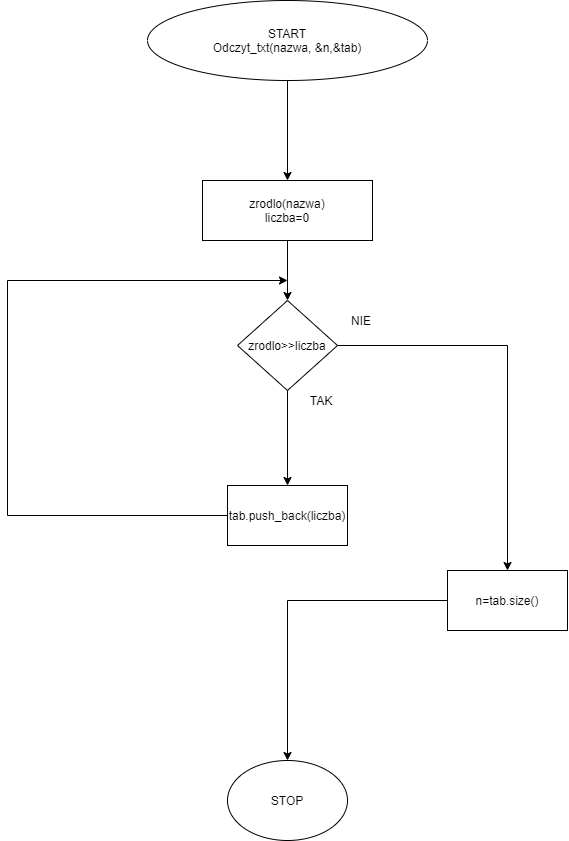
Gdy podczas przeszukiwania tablicy, mającego na celu znalezienie maksymalnego iloczynu liczb dodatnich, zostanie znaleziona liczba ujemna to zapisywana jest ona do oddzielnej tablicy (wektora). Dzięki temu przy poszukiwaniu największego iloczynu liczb ujemnych możemy sprawdzać tablicę o mniejszych wymiarach od wyjściowej. Podzielenie problemu na dwie osobne tablice sprawia, że nie są wymnażane przez siebie liczby dodatnie z ujemnymi, które to dają ujemny iloczyn. Maksymalny iloczyn będący liczbą ujemną może wystąpić jedynie dla tablicy dwuelementowej, dlatego w przypadku większych tablic nie trzeba rozważać takiego przypadku.

Kolejną zastosowaną optymalizacją jest porównywanie do siebie wartości dwóch wymnażanych liczb, w przypadku gdy wartość bezwzględna drugiej liczby jest większa bądź równa wartości bezwzględnej pierwszej liczby to pierwsza liczba nie jest dalej wykorzystywana przy szukaniu maksymalnego iloczynu. W takim przypadku w dalszych poszukiwaniach korzystamy już z drugiej liczby.

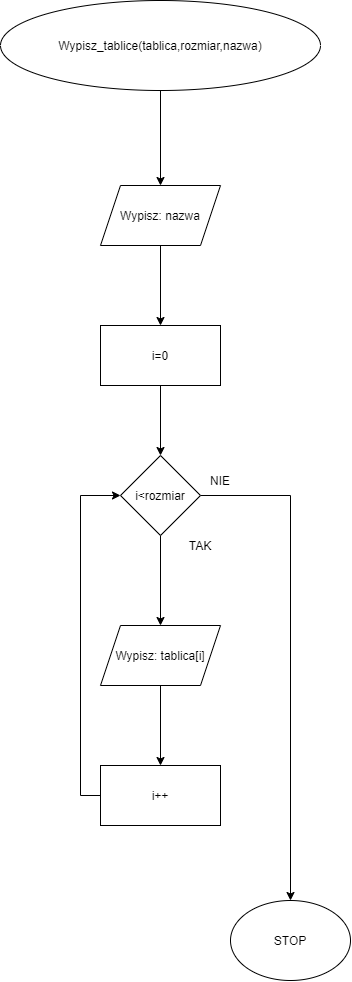
# Schematy blokowe



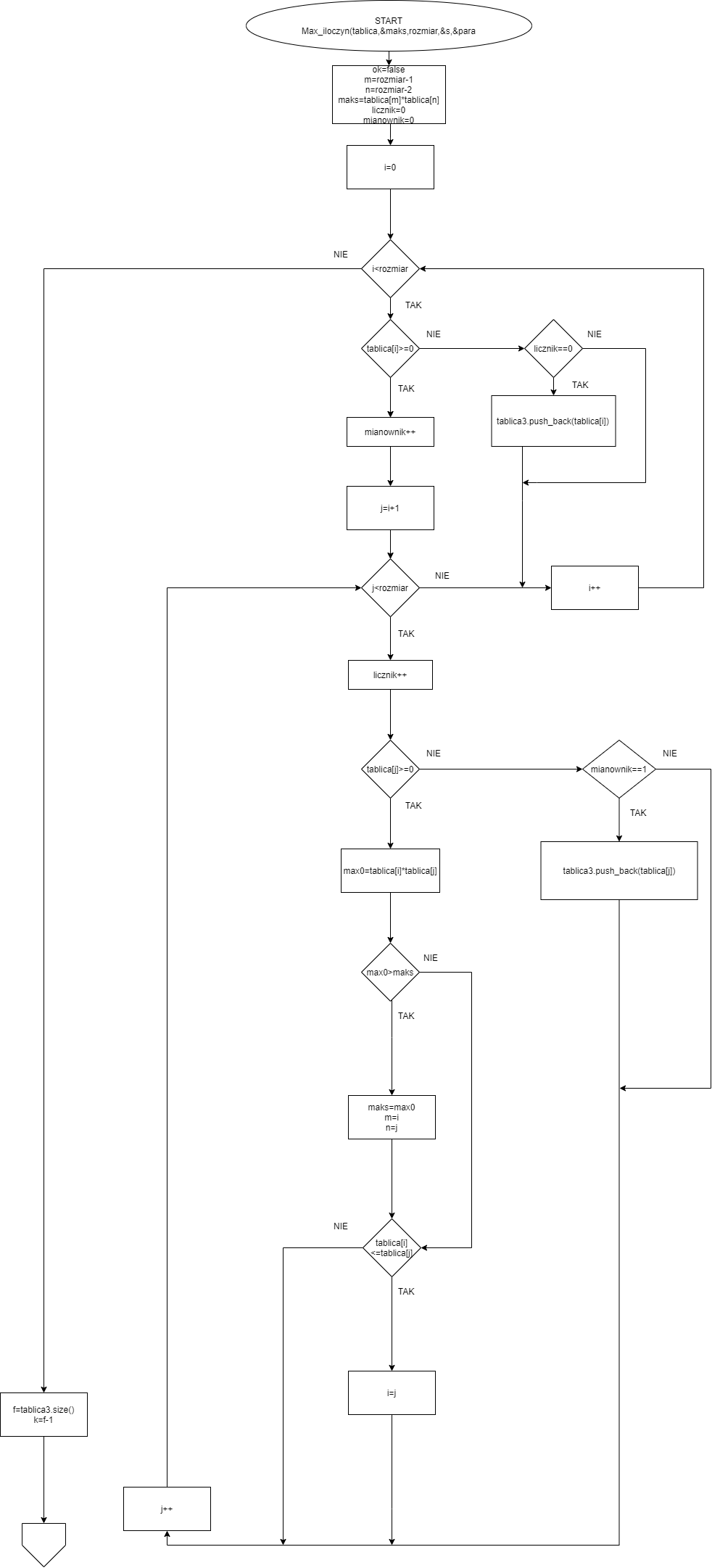
Rysunek .. Schemat blokowy programu

****

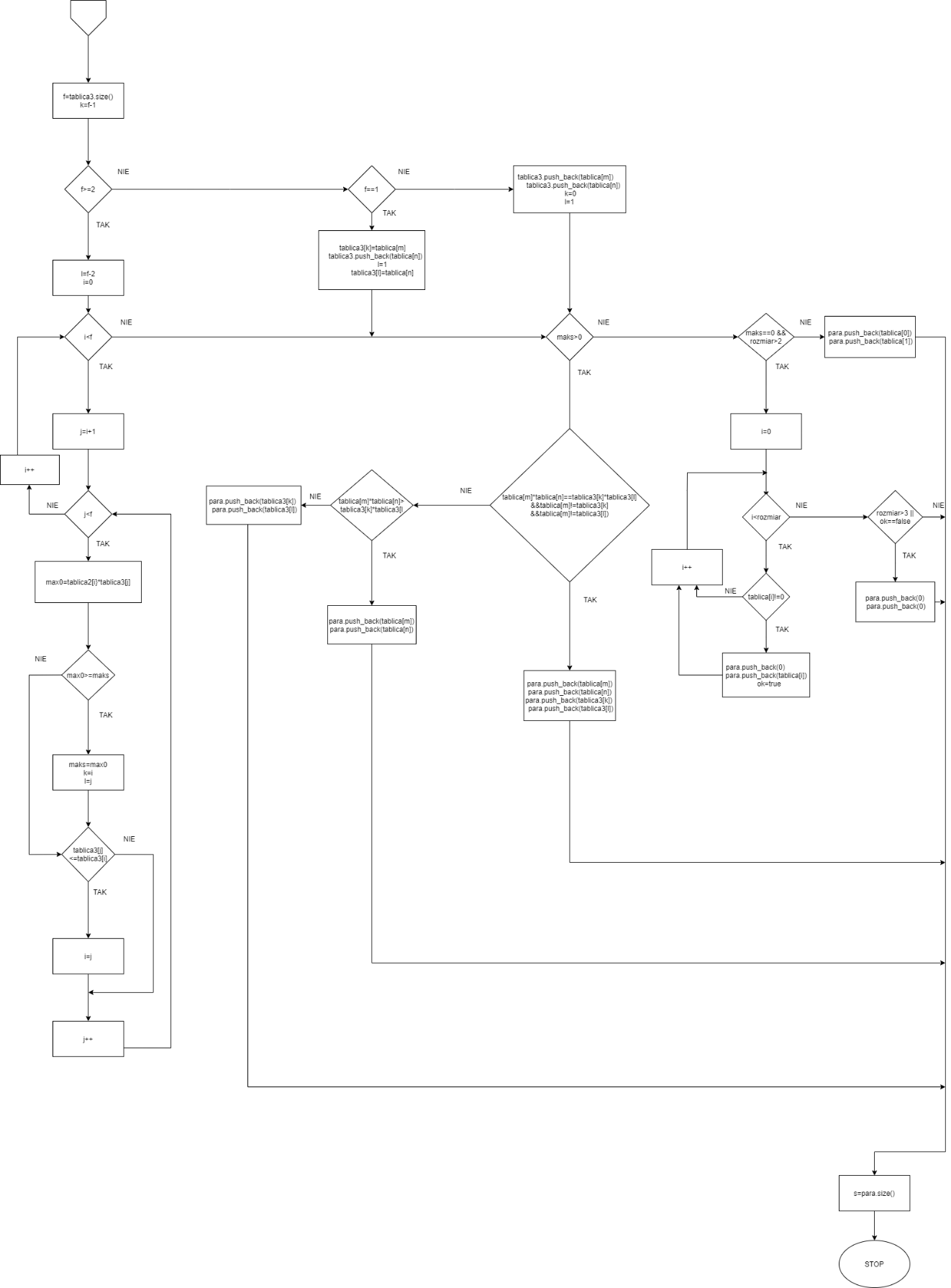
Rysunek .. Schemat blokowy funkcji Odczyt\_txt

****

Rysunek .. Schemat blokowy funkcji Wypisz\_tablice

****

Rysunek .. Schemat blokowy funkcji Max\_iloczyn cz.1

****

Rysunek .. Schemat blokowy funkcji Max\_iloczyn cz.2

# Pseudokod

**Main**

wywołaj funkcję Odczyt\_txt(nazwa pliku, liczba elementów n, wektor v)

wywołaj funkcję Wypisz tablice(wektor v, liczba elementów n, nazwa tablicy)

jeśli liczba elementów n>=2

wywołaj funkcję Max\_iloczyn(wektor v,max1,liczba elementów n,

liczba elementów k, wektor A)

wywołaj funkcję Zapisz(nazwa pliku,max1,liczba elementów k, wektor A)

jeśli nie

wyświetl komunikat o braku wystarczającej liczby elementów w tablicy

zakończ

**Odczyt txt (nazwa ,n, tab)**

otwórz plik nazwa

liczba=0

dopóki w pliku jest liczba, wykonuj:

dopisz zmienną liczba do wektora tab

przypisz zmiennej n długość wektora tab

zakończ

**Wypisz\_tablice(tablica, rozmiar, nazwa)**

wypisz nazwa

począwszy od i=0, dopóki i<rozmiar, wykonuj:

wypisz tablica[i]

zwiększ i o 1

zakończ

**Zapisz(nazwa,maks,k,A)**

utwórz plik o zadanej nazwie

przekaż element maks do pliku

począwszy od i=0, dopóki i<k, wykonuj:

przekaż element A[i] oraz A[i+1] do pliku

zwiększ i o 2

zakończ

**Max\_iloczyn(tablica,maks,rozmiar,s,para)**

mianownik=0

licznik=0

m=rozmiar-1

n=rozmiar-2

ok=false

maks=tablica[m]\*tablica[n]

począwszy od i=0, dopóki i<rozmiar, wykonuj:

jeśli tablica[i]>=0

zwiększ mianownik o 1

począwszy od j=i+1, dopóki j<rozmiar, wykonuj:

zwiększ licznik o 1

jeśli tablica[j]>=0

max0=tablica[i]\*tablica[j]

jeśli max0>maks

maks=max0

m=i

n=j

jeśli tablica[i]<=tablica[j]

i=j

zwiększ j o 1

jeśli nie i mianownik==1

dodaj element tablica[j] do wektora tablica3

jeśli nie i licznik==1

dodaj element tablica[i] do wektora tablica3

zwiększ i o 1

przypisz zmiennej f wartość długości wektora tablica3

k=f-1

jeśli f>=2

l=f-2

począwszy od i=0, dopóki i<f, wykonuj:

począwszy od j=i+1, dopóki j<f, wykonuj:

max0=tablica3[i]\*tablica3[j]

jeśli max0>=maks

maks=max0

k=i

l=j

jeśli tablica3[j]<=tablica3[i]

i=j

zwiększ j o 1

zwiększ i o 1

jeśli nie oraz f==1

tablica3[k]=tablica[m]

dodaj element tablica[n] do wektora tablica3

l=1

tablica3[l]=tablica[n]

jeśli nie

dodaj element tablica[m] do wektora tablica3

dodaj element tablica[n] do wektora tablica3

k=0

l=1

jeśli maks>0

jeśli tablica[m]\*tablica[n]==tablica3[k]\*tablica3[l] i wartość tablica[m] nie jest równa tablica3[l] i wartość tablica[m] nie jest równa tablica3[l], to

do wektora para dodaj elemnty:

tablica[m], tablica[n], tablica3[k], tablica3[l]

jeśli nie i tablica[m]\*tablica[n]>tablica3[k]\*tablica3[l]

do wektora para dodaj elementy tablica[m],tablica[n]

jeśli nie

do wektora para dodaj elementy tablica3[k],tablica3[l]

jeśli nie oraz maks==0 i rozmiar>2

począwszy od i=0, dopóki i<rozmiar

jeśli tablica[i] nie jest równa 0

do wektora para dodaj element 0 oraz tablica[i]

ok=true

jeśli rozmiar>3 lub ok==false

do wekotra para dodaj dwa elementy o wartości 0

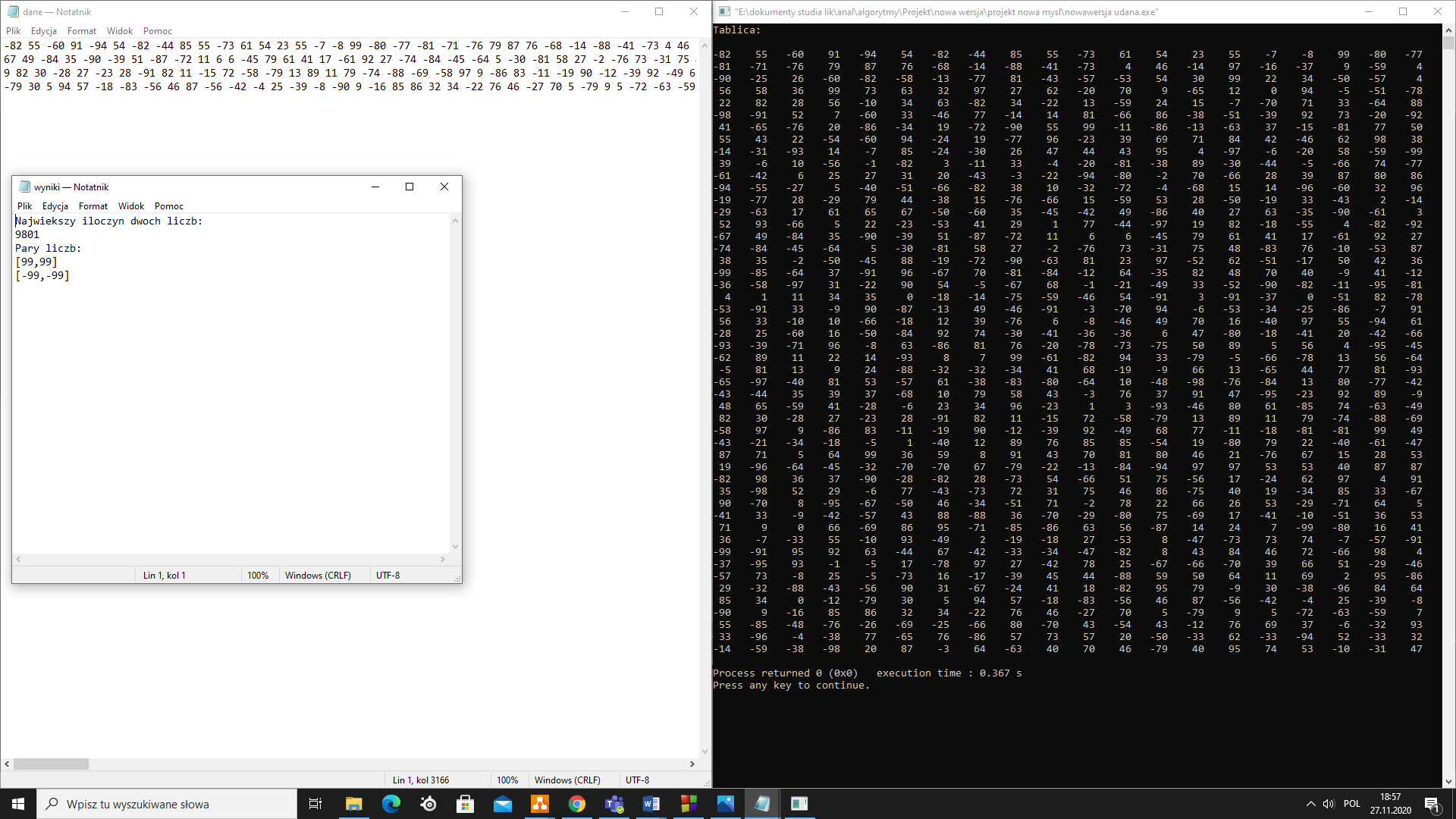
jeśli nie

do wektora para dodaj element tablica[0] oraz tablica[1]

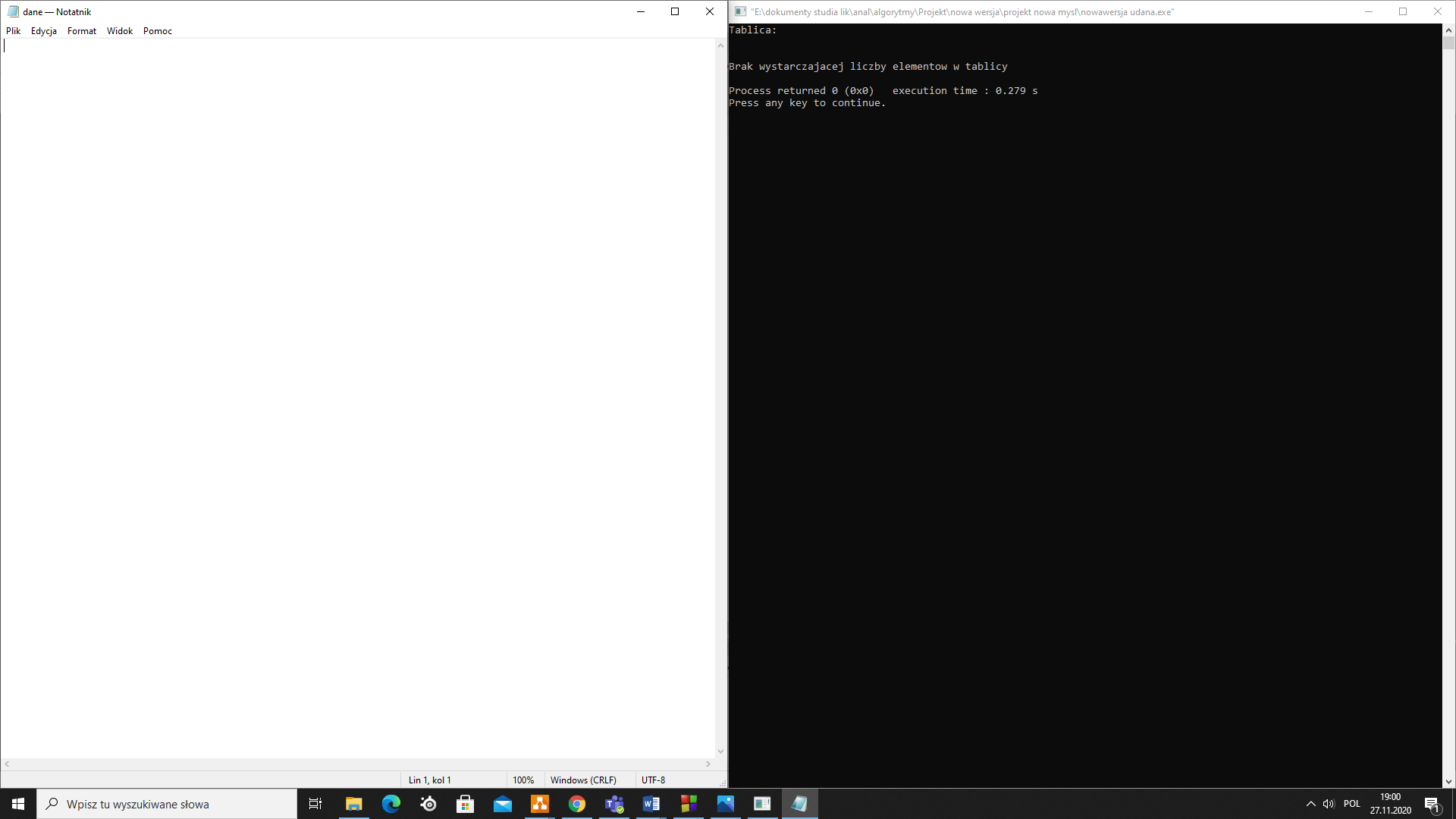
przypisz zmiennej s długość wektora para

zakończ

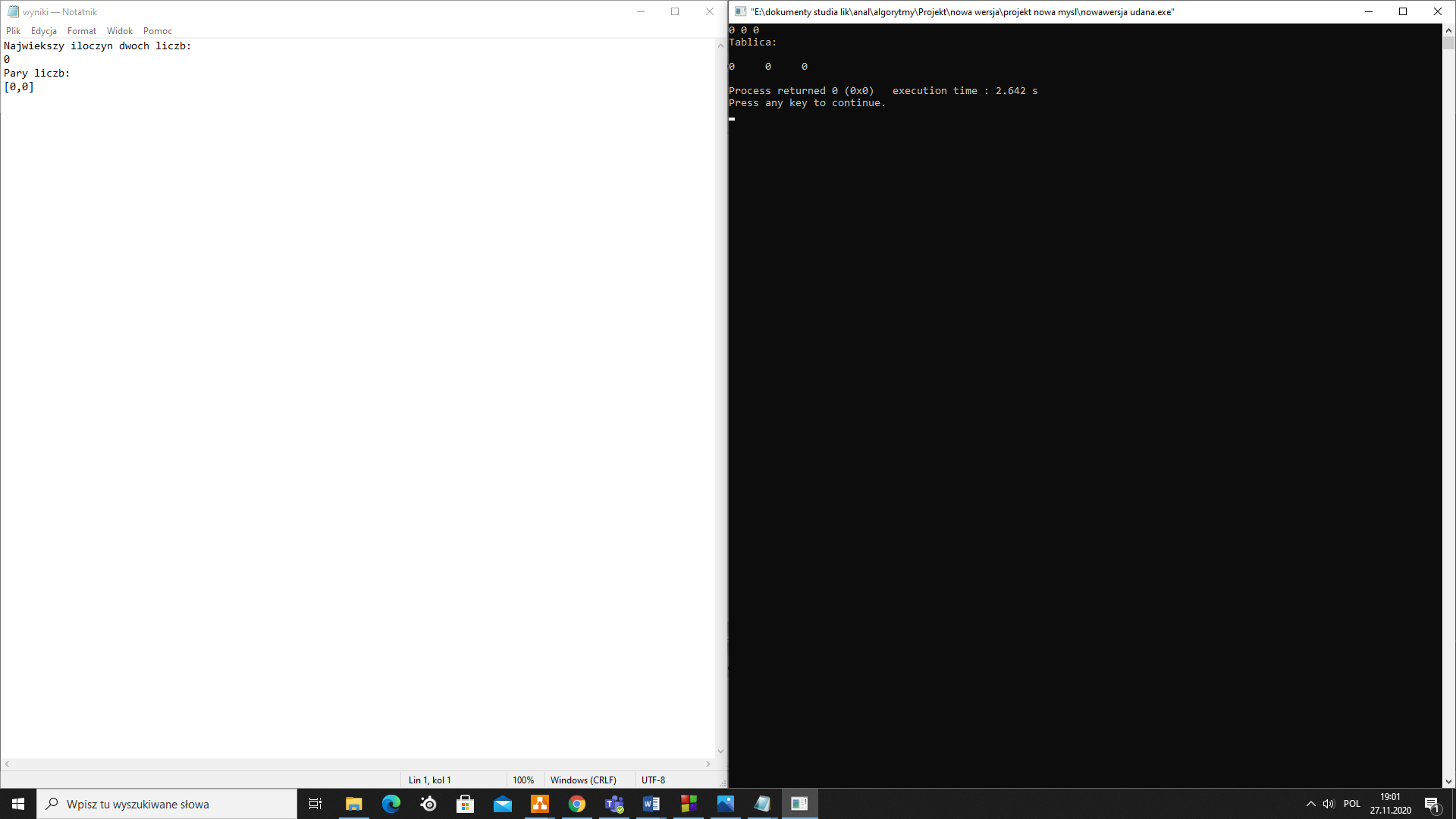
# Dokumentacja z doświadczeń



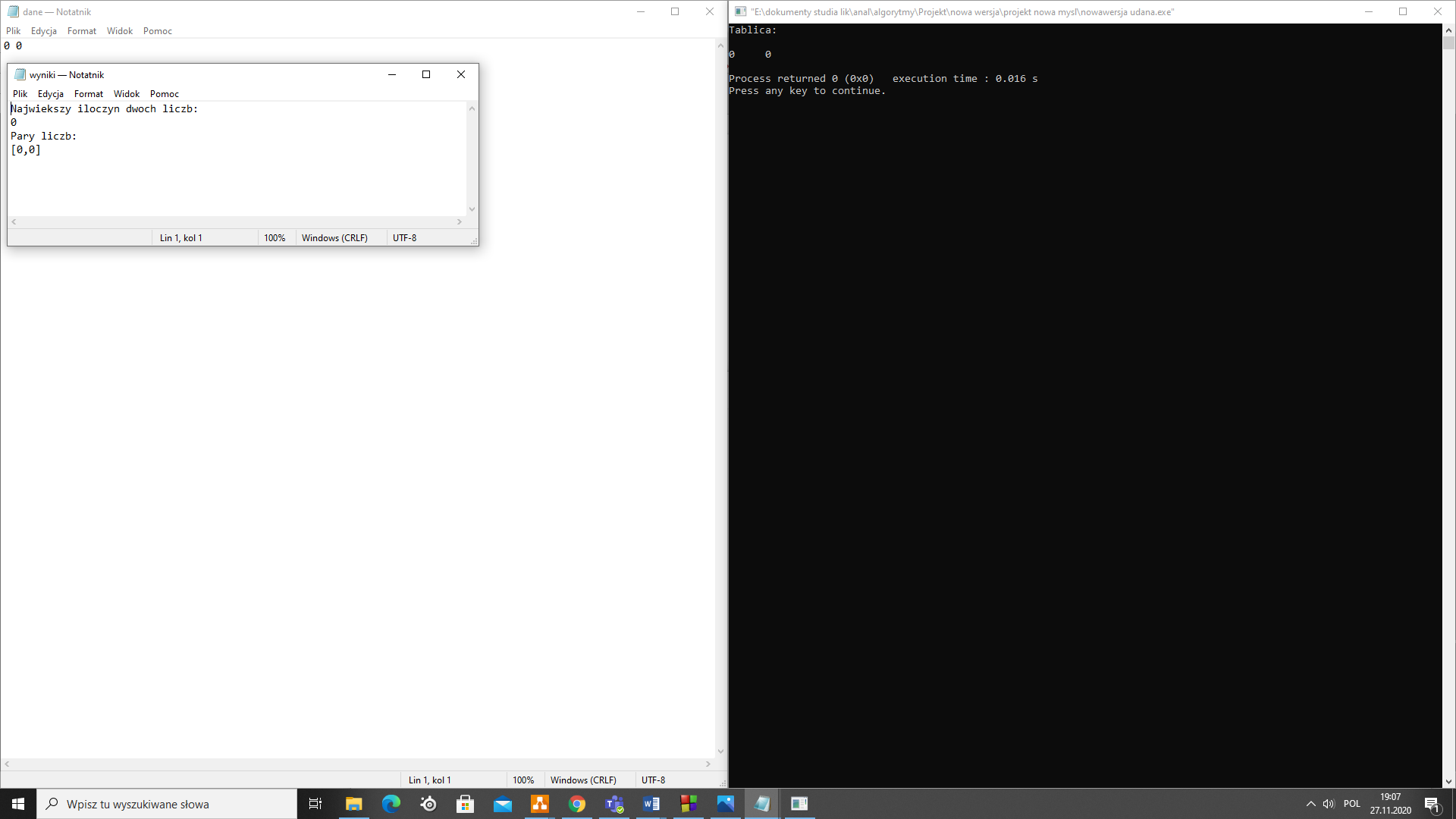
Rysunek .. Działanie programu dla 1000 liczb w tablicy z zakresu od -99 do 99



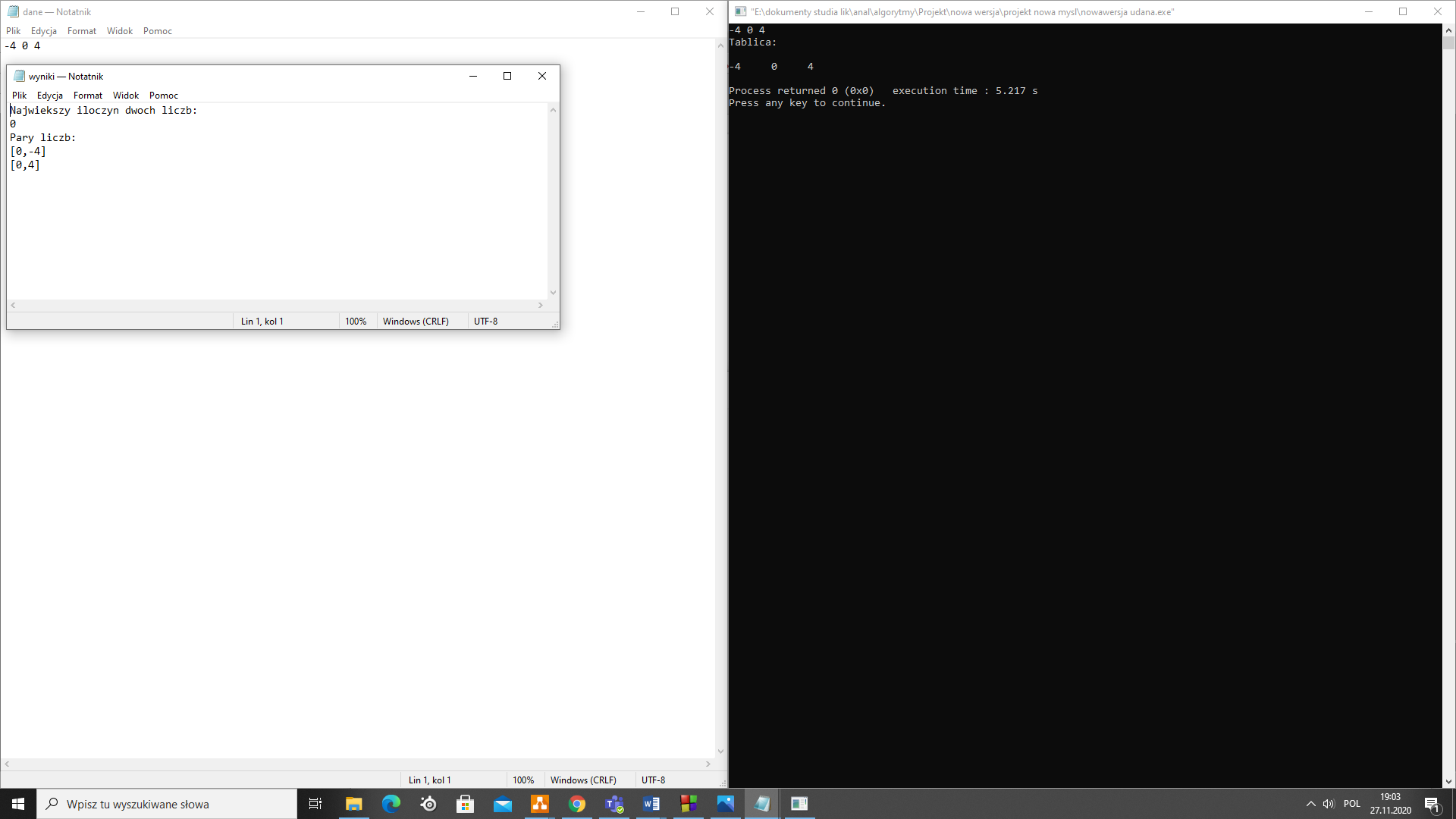
Rysunek .. Działanie programu w przypadku braku liczb wejściowych w pliku



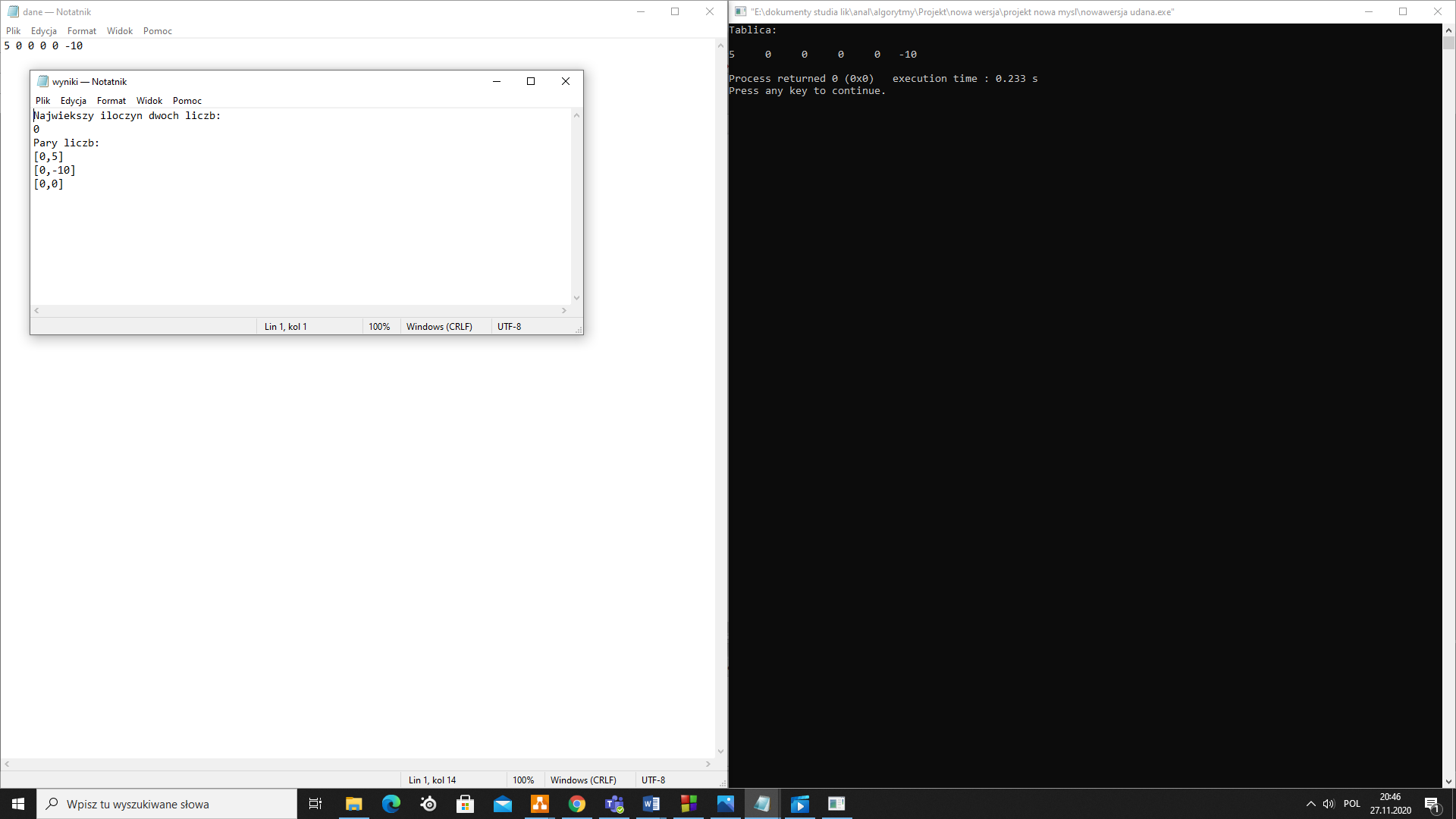
Rysunek .. Działanie programu dla trzech elementów równych 0



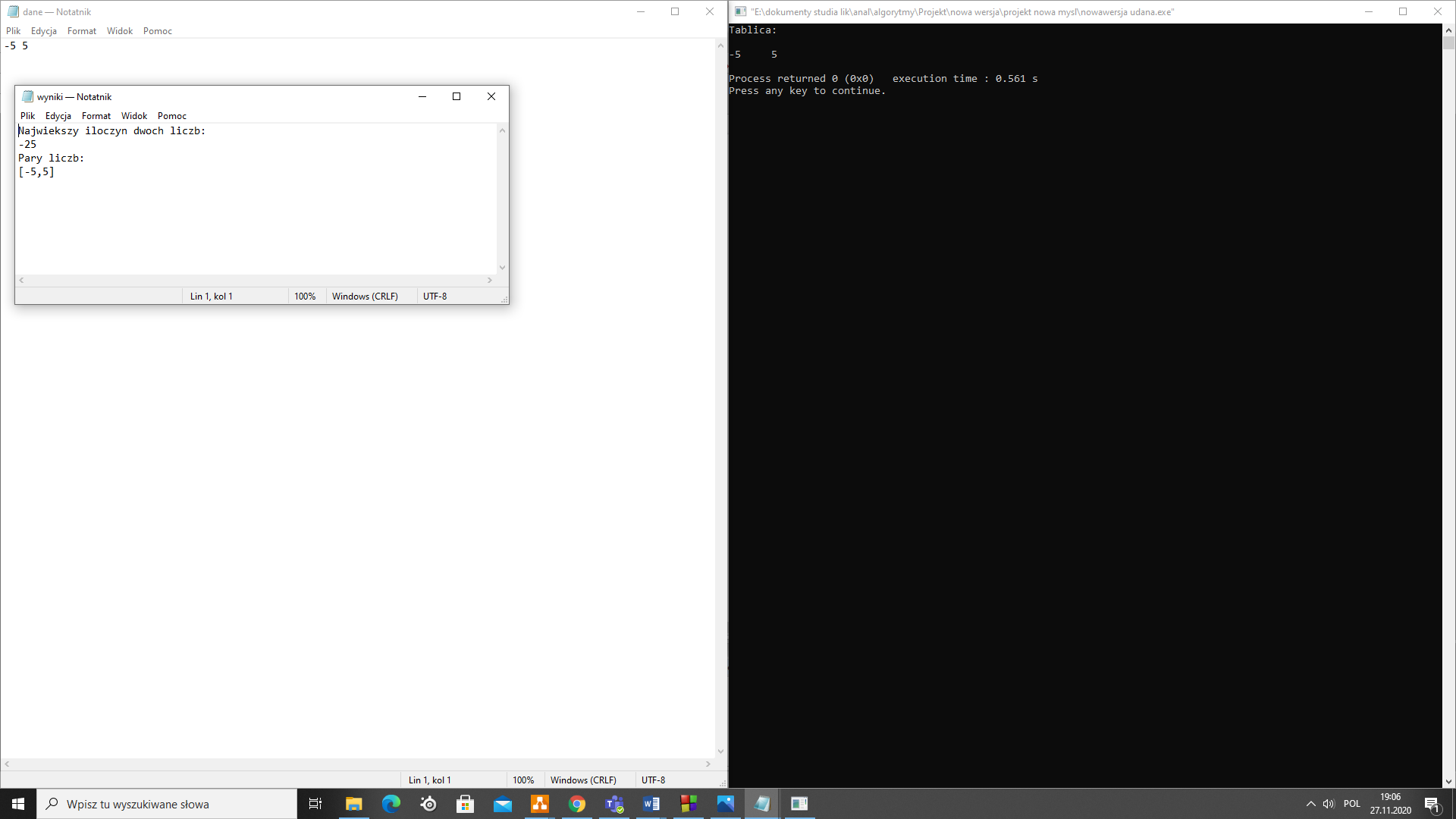
Rysunek .. Działanie programu dla dwóch zerowych elementów



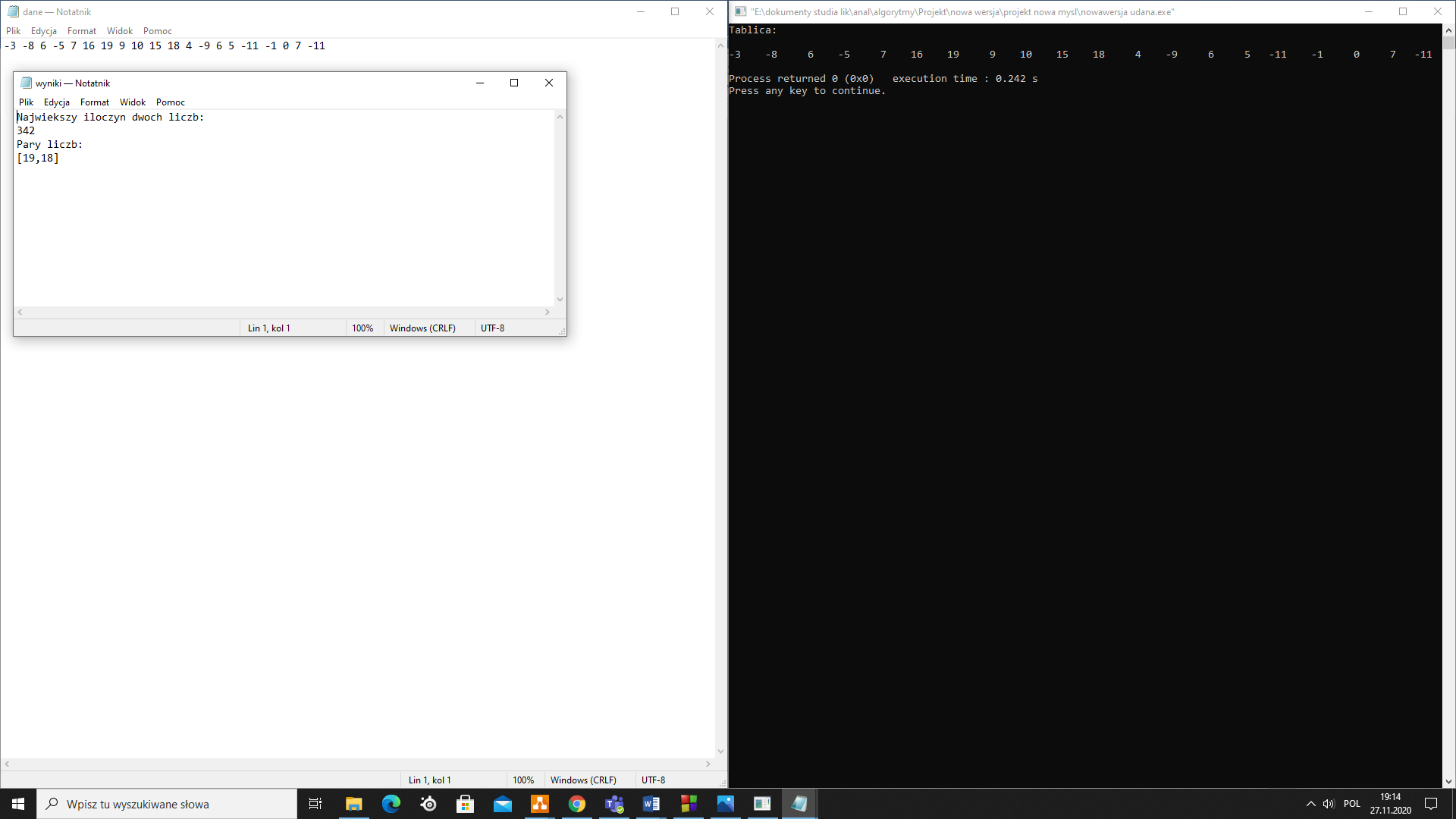
Rysunek .. Działanie programu dla 3 elementów z największym iloczynem równym 0



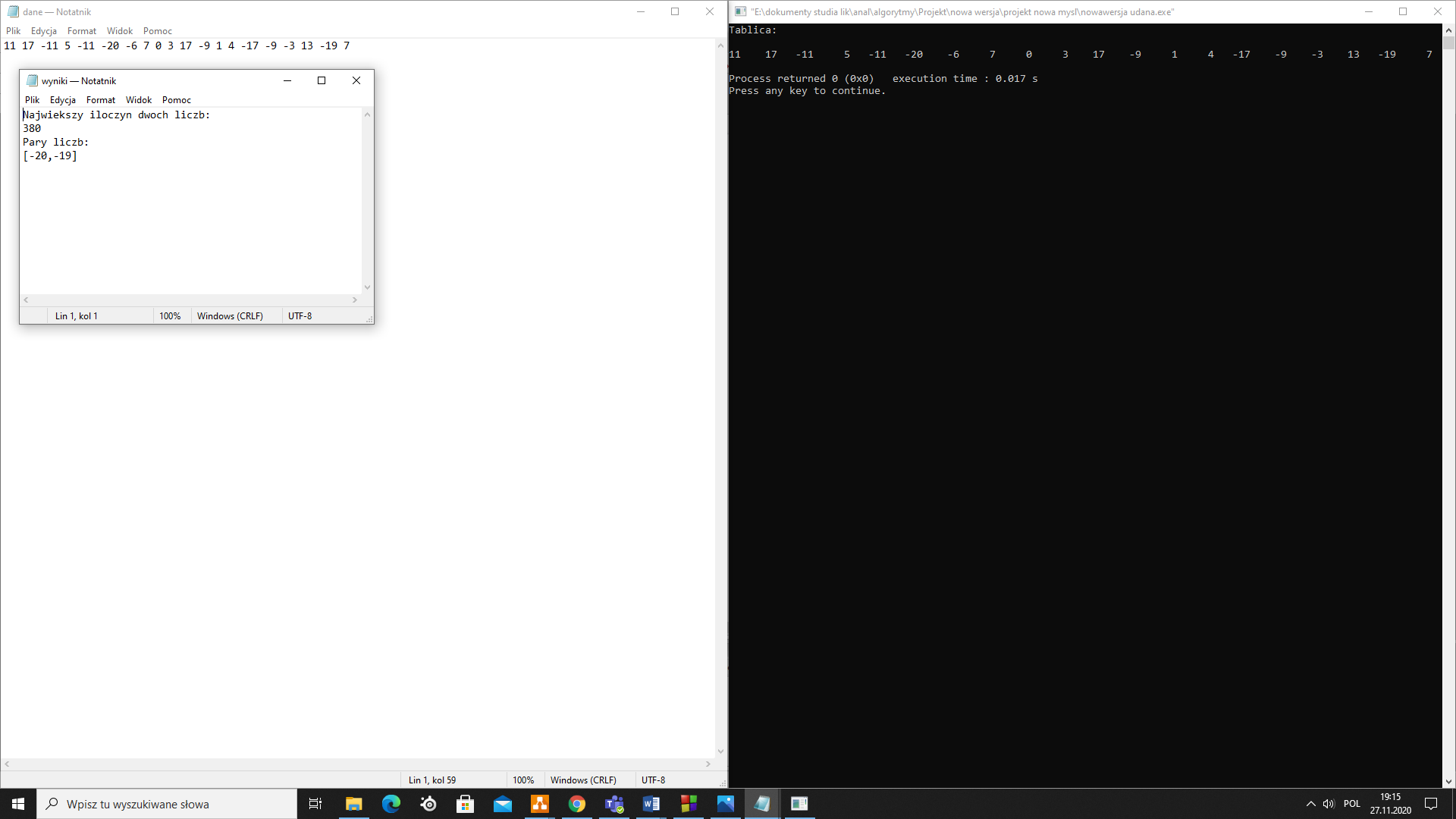
Rysunek .. Działanie programu dla 5 elementów z największym iloczynem równym 0



Rysunek .. Działanie programu dla największego iloczynu mniejszego od 0



Rysunek .. Działanie programu dla 20 liczb z zakresu od -20 do 20, przypadek 1



Rysunek .. Działanie programu dla 20 liczb z zakresu -20 do 20, przypadek 2

# Wnioski

Podsumowując, warto powiedzieć, że zrealizowano cel projektu. Stworzono program wyznaczający maksymalny iloczyn dwóch liczb. Program umożliwia odczyt liczb z pliku tekstowego, wykonuje obliczenia a następnie zapisuje do pliku tekstowego maksymalny iloczyn oraz liczby go wyznaczające.

Na podstawie przeprowadzonych testów widoczne jest, że program działa poprawnie dla analizowanych przypadków. Uwzględnia on prawidłowo przypadki szczególne, w których maksymalny iloczyn jest równy 0.

W celu poprawy programu należałoby zastanowić się nad uproszczeniem zapisu kodu oraz nad wcześniejszym przerwaniem pętli for przy wyszukiwaniu maksymalnego iloczynu.

# Kod programu

#include <iostream>

#include <fstream>

#include<time.h>

#include<cstdlib>

#include <vector>

#include <iomanip>

using namespace std;

void Wprowadzanie(const char\*nazwa) //funkcja do wprowadzania danych

void Odczyt\_txt(const char\* nazwa,int &n,vector<int>&tab)

{

ifstream zrodlo(nazwa);

int liczba=0;

while(zrodlo>>liczba) //dopoki w zrodle znajduje sie liczba

tab.push\_back(liczba); //dodaj element do wektora tab

n=tab.size(); //rozmiar wektora tab

}

void Wypisz\_tablice(vector<int> tablica, int rozmiar, const char\*nazwa)

{ //funkcja do wypisywaania tablicy

int i;

cout<<nazwa<<":"<<endl<<endl;

for (i=0;i<rozmiar;i++)

cout<<tablica[i]<<setw(4);

cout<<endl;

}

void Max\_iloczyn(vector<int> tablica,int & maks,int rozmiar,int& s, vector<int> &para)

{

int i,j;

int max0;

int m,n,k,l;

int f;

int mianownik=0; //niefortunna nazwa zmiennej, sluzy do zliczania jak licznik

int licznik=0;

bool ok=false;

m=rozmiar-1;

n=rozmiar-2;

vector <int> tablica3;

maks=tablica[m]\*tablica[n]; //przypisanie maksymalnego iloczynu do dwoch ostatnich elementow tablicy

for(i=0;i<rozmiar;i++)

{

if(tablica[i]>=0) //sprawdzanie czy dodatnia

{ mianownik++;

for(j=i+1;j<rozmiar;j++)

{licznik++;

if(tablica[j]>=0) //sprawdzanie czy dodatnia

{

max0=tablica[i]\*tablica[j]; //przypisanie maksa do kolejnych elementow

if(max0>maks) //sprawdzenie czy nowy iloczyn jest wiekszy od poprzedniego najwiekszego

{

maks=max0;

m=i; //przypisanie w celu latwego znalezienia liczb wyznaczajacych najwiekszy iloczyn

n=j;

}

if(tablica[i]<=tablica[j])

i=j; //przypisanie w celu nie sprawdzania dalszych elementow tablicy

// przy pomocy liczby ktora nie da najwiekszego iloczynu

}

else if (mianownik==1) //sprawdzenie warunku zeby nie dodac dwa razy tych samych liczb

tablica3.push\_back(tablica[j]);

}

}

else if (licznik==0) //sprawdzenie warunku jak wyzej

tablica3.push\_back(tablica[i]);

}

f=tablica3.size(); //rozmiar tablicy liczb ujemnych

k=f-1;

if (f>=2)

{l=f-2;

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

for(int j=i+1;j<f;j++)

{

max0=tablica3[i]\*tablica3[j];

if(max0>=maks) //analogicznie jak poprzedniu ale zamiast"=" mamy ">=" umozliwia znaleznienie

{ maks=max0; //liczb w przypadku gdy iloczyn liczb dodatnich jest rowny iloczynowi liczb ujemnych

k=i;

l=j;

}

if(tablica3[j]<=tablica3[i])

i=j;

}

}

else if (f==1) //gdy rozmiar tablicy liczb ujemnych jest rowny 1, dopisujemy element

{

tablica3[k]=tablica[m];

tablica3.push\_back(tablica[n]);

l=1;

tablica3[l]=tablica[n];

}

else //gdy brak elementow, dopisujemy dwa elementy

{

tablica3.push\_back(tablica[m]);

tablica3.push\_back(tablica[n]);

k=0;l=1;

}

if(maks>0)

{

if(tablica[m]\*tablica[n]==tablica3[k]\*tablica3[l]&&tablica[m]!=tablica3[k]&&tablica[m]!=tablica3[l]) //gdy maks dodatnich jest

//rowny maksowi ujemnych

{

para.push\_back(tablica[m]);

para.push\_back(tablica[n]);

para.push\_back(tablica3[k]);

para.push\_back(tablica3[l]);

}

else if(tablica[m]\*tablica[n]>tablica3[k]\*tablica3[l]) //gdy maksymalny iloczyn dodatnich wiekszy od maksymalnego ujemnych

{

para.push\_back(tablica[m]);

para.push\_back(tablica[n]);

}

else //gdy maksymalny iloczyn liczb ujemnych jest wiekszy niz liczb dodatnich

{

para.push\_back(tablica3[k]);

para.push\_back(tablica3[l]);

}

}

else if(maks==0 && rozmiar>2) //gdy tablica z samymi zereami i ewnetualnie jedna liczba dodania i/lub ujemna

{

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

if(tablica[i]!=0)

{

para.push\_back(0); //dodanie do wektora wyjsciowego pary 0 oraz niezerowej liczby

para.push\_back(tablica[i]);

ok=true;

}

if(rozmiar>3 || ok==false) // gdy rozmiar tablicy wiekszy od 3, maks iloczyn daje tez para [0,0]

{ //gdy mamy same 0 w tablicy 3-elementowej chcemy wypisac pare [0,0] tylko raz

para.push\_back(0);

para.push\_back(0);

}

}

else //gdy mamy dwa elementy z maksymalnym iloczynem mniejszym od 0 (dodatnia i ujemna)

{

para.push\_back(tablica[0]);

para.push\_back(tablica[1]);

}

s=para.size(); //rozmiar wektora

}

void Zapisz(const char\*nazwa,int maks, int k, vector<int>A)

{

ofstream plik(nazwa);

plik<< "Najwiekszy iloczyn dwoch liczb:"<<endl;

plik << maks<<endl;

plik<<"Pary liczb:"<<endl;

for(int i=0;i<k;i+=2) //wypisanie par liczb z wektora wyznaczonego w funkcji Max\_iloczyn

plik<<"["<<A[i]<<","<<A[i+1]<<"]"<<endl;

}

int main()

{

int n,k;

vector <int> v;

vector<int>A;

int max1;

// srand ( ( unsigned )time ( NULL ) );

// Wprowadzanie("dane.txt");

Odczyt\_txt("dane.txt",n,v);

Wypisz\_tablice(v,n,"Tablica");

if(n>=2)

{

Max\_iloczyn(v,max1,n,k,A);

Zapisz("wyniki.txt",max1,k,A);

}

else

cout<<"Brak wystarczajacej liczby elementow w tablicy"<<endl;

return 0;

}