2021

Inżynieria i analiza
danych 2021/2022

Mateusz Sokal

[SPRAWOZDANIE -PROJEKT 12.11.2021]

Projekt nr 1, Grupa projektowa: P07

Opis problemu, podstawy teoretyczne i implementacja

Treść: Dla zadanej tablicy liczb całkowitych znajdź maksymalny iloczyn dwóch elementów znajdujących się w tej tablicy.

Przykład:

Wejście: A = [-10,8,5,-4,1]

Wyjście: Czynniki generujące maksymalny iloczyn to: [-10,-4] i [8,5]

Tematem tego projektu było znalezienie maksymalnego iloczynu wśród n-elementowego zbioru liczb całkowitych.

Tablica to zbiór danych tego samego typu. Tablicę, tak samo jak pojedyncze zmienne deklarujemy według wzoru:

typ_danych nazwa_tablicy [ilość_elementów]

Jeżeli znamy liczbę, i wartość elementów tablicy, zamiast **[ilość_elementów]** możemy je wypisać jako:

typ_danych nazwa_tablicy = {lista_elementów_tablicy}

Kolejną funkcją potrzebną do wykonania zadania jest **pętla for,** która umożliwia powtarzanie pewnych instrukcji do momentu spełnienia warunku końcowego. Pętlę tą definiujemy w ten sposób:

for (typ_danych wyrażenie1, wyrażenie2, wyrażenie3)

wyrażenie1 - instrukcja wykonana przed pierwszym obiegiem pętli,

wyrażenie2 - warunek zakończenia pętli,

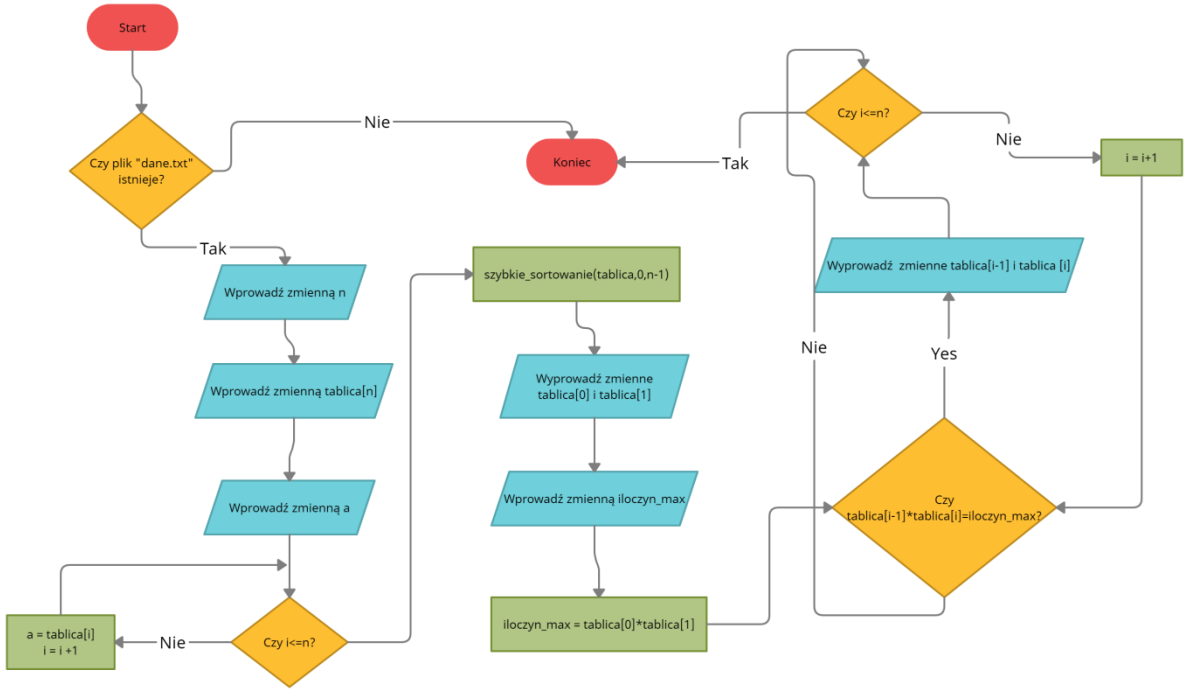
wyrażenie3 - opis instrukcji wykonywanej za każdym obiegiem pętli.

Pętla while wykonuje instrukcję dopóki warunek końcowy jest spełniony, od **pętli for** różni się tym, że nie trzeba definiować po ilu powtórzeniach pętla ma się zakończyć. **Pętlę while** definiujemy jako:

while(warunek)

{instrukcja}

Schemat blokowy algorytmu



Źródło: app.creately.com

Algorytm w pseudokodzie

Wczytaj n

Wczytaj a

Inicjuj tablica[n]

Jeżeli plik dane.txt **nie istnieje**

Zakończ program

W przeciwnym razie

Dla i=0 do i<n **powtarzaj**

Wczytaj a **z pliku**

tablica[i]=a

Sortuj dane malejąco z tablica[n]

Wczytaj iloczyn_max

I iloczyn_max = tablica[0]*tablica[1]

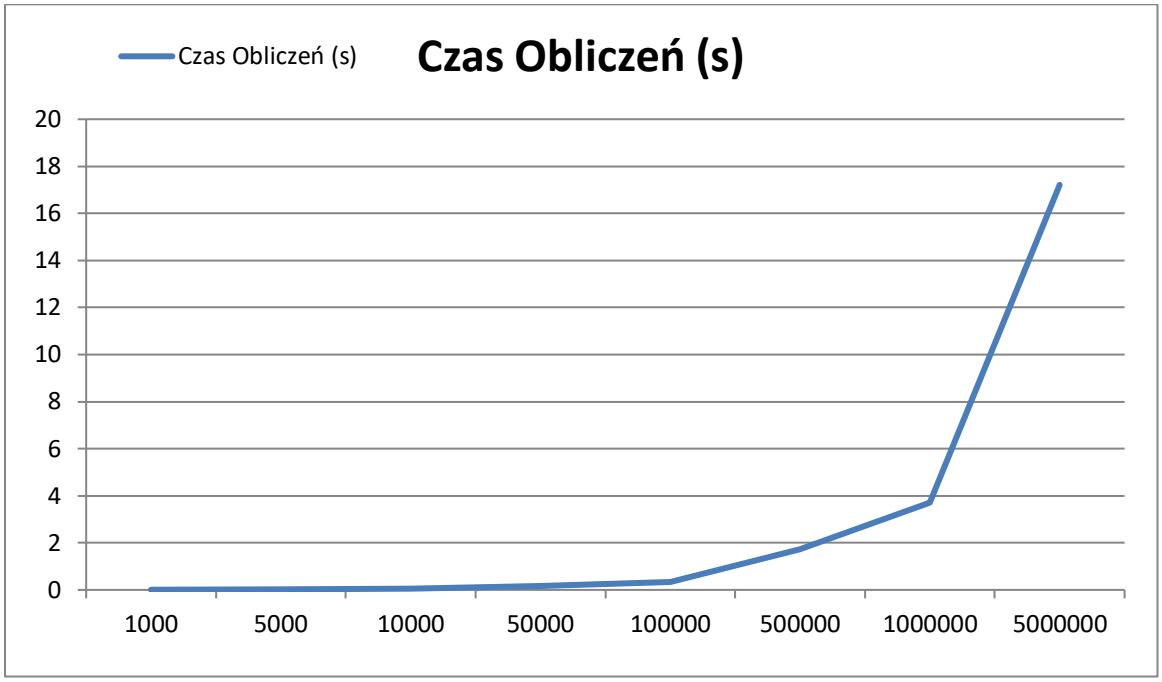
Dla i=0 do i<n **powtarzaj**

Jeżeli tablica[n]*tablica[n-1]=iloczyn_max

Wypisz tablica[n], tablica[n-1]

Zakończ program

Wykres czasu obliczeń algorytmu



Z wykresu można wywnioskować, że złożoność czasowa algorytmu wynosi O(2n) a złożoność obliczeniowa jest równa n^2.

Kod programu

```
#include <iostream>

#include <fstream>

#include <time.h>

#include <cstdlib>

using namespace std;


void quicksort(int *tablica, int lewy, int prawy)

{

   int v=tablica[(lewy+prawy)/2];

   int i,j,x;

   i=lewy;
```

```
    j=prawy;
    do
    {
        while(tablica[i]>v) i++;
        while(tablica[j]<v) j--;
        if(i<=j)
        {
            x=tablica[i];
            tablica[i]=tablica[j];
            tablica[j]=x;
            i++;
            j--;
        }
    }
    while(i<=j);
    if(j>lewy) quicksort(tablica,lewy, j);
    if(i<prawy) quicksort(tablica, i, prawy);
}

int main()
{
    fstream dane, wyniki;
    int *tab, n;
    double a, czas;
    long iloczynMax;
    clock_t start, stop;

    dane.open("dane.txt", ios::in);
    wyniki.open("wyniki.txt", ios::out);
    if(dane.good()==false)
    {
        cout << "Plik dane.txt nie istnieje! Dolacz plik do folderu projektu i sprobuj ponownie.";
        return 0;
    }
```

```
    cout<<"Ile liczb znajduje sie w tablicy? ";
    cin>>n;

    tab = new int [n];

    start = clock();
    for(int i=0; i<n; i++)
    {
       dane>>a;
       tab[i]=a;
    }

    quicksort(tab, 0, n-1);

    cout << endl;
    iloczynMax = tab[0] * tab[1];
    cout << "Maksymalny iloczyn: " << iloczynMax << endl;

    cout << "Pary liczb generujace najwiekszy iloczyn: " << endl;
    wyniki << "Pary liczb generujace najwiekszy iloczyn: " << endl;
    for (int i=1; i<n; i++)
    {
       if (tab[i-1]*tab[i]==iloczynMax)
       {
          cout << tab[i-1] << " " << tab[i] << endl;
          wyniki << tab[i-1] << " " << tab[i] << endl;
       }
    }
    stop = clock();
    czas = (double)(stop - start)/CLOCKS_PER_SEC;
    cout << "Czas pracy programu: " << czas << endl;
    return 0;
}
```