

SPRAWOZDANIE

Paweł Dziejczak P02

Projekt 1

Spis treści

Wstęp.....	1
Treść zadania.....	1
Opis problemu	1
Schemat blokowy	2
Pseudokod	3
Działanie kodu.....	4
Szkielet kodu	5

Wstęp

Dokument jest sprawozdaniem z pierwszego projektu z przedmiotu Algorytmy i struktury danych. Program został napisany w języku C++ w kompilatorze Visual Studio 2019

Treść zadania

Dla ciągu (w postaci tablicy) zawierającego wyłącznie wartości 0 lub 1, znajdź podciąg zawierający równą liczbę zer i jedynek, którego długość jest największa

Przykład.

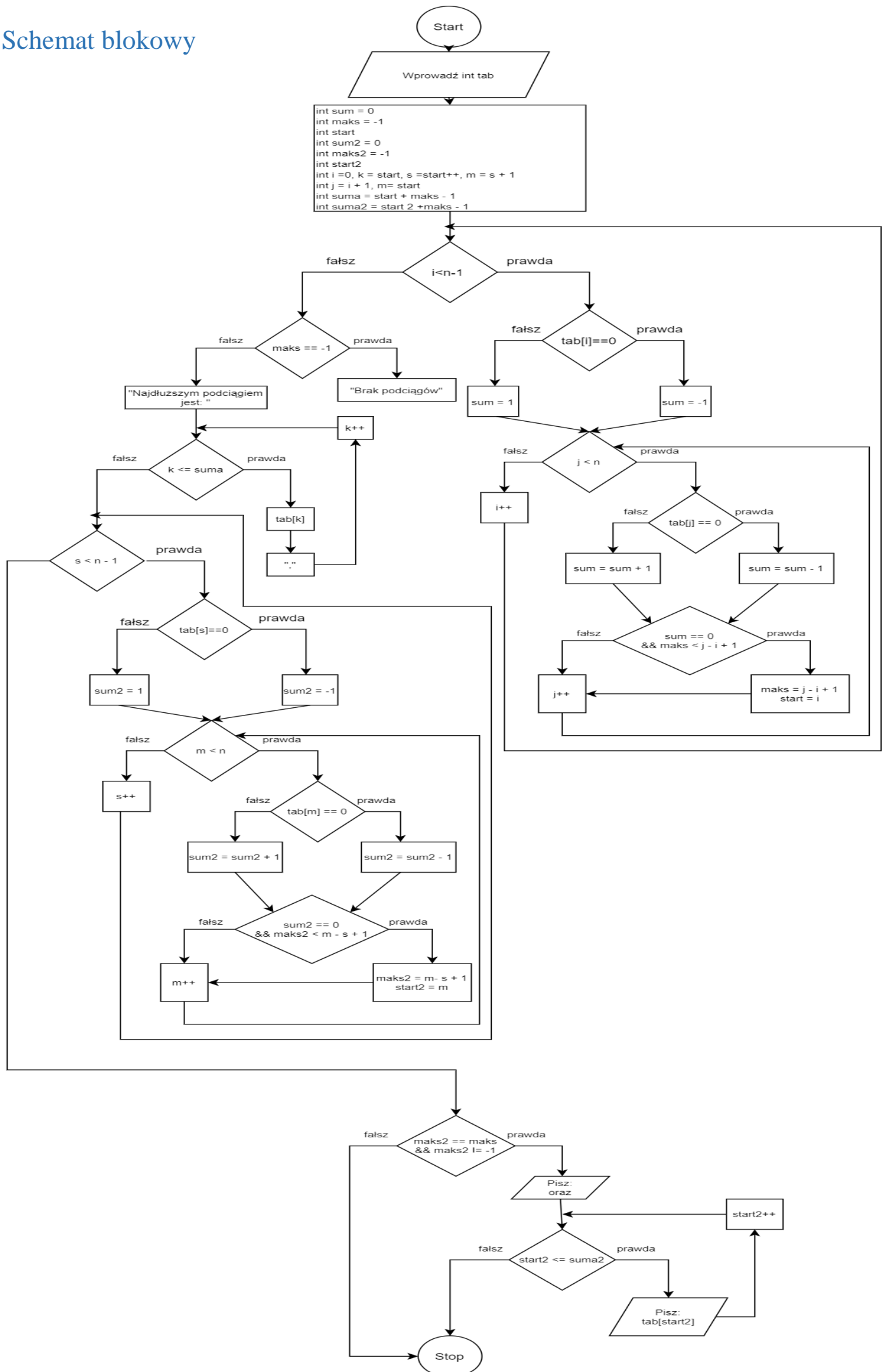
Wejście: 0,0,1,0,1,0,0

Wyjście: Najdłuższym podciągiem są: 0,1,0,1 oraz 1,0,1,0

Opis problemu

Aby znaleźć najdłuższy podciąg zer i jedynek, posłużyłem się metodą zamiany każdego zera w głównym podciągu na -1 a następnie podliczenia sumy każdego następnego wyrazu w tym ciągu. W wyniku czego, gdy suma wyrazów wynosi 0, początek i koniec naszego podciągu jest zapisywany tym samym wyznaczając nam początek i koniec najdłuższego aktualnego podciągu. W przypadku, gdy w głównym ciągu znajduje się inny podciąg spełniający warunek o równej ilości zer i jedynek, jest on porównywany z dotychczasowym podciągiem i nadpisywany jeśli jego długość jest większa od poprzedniego.

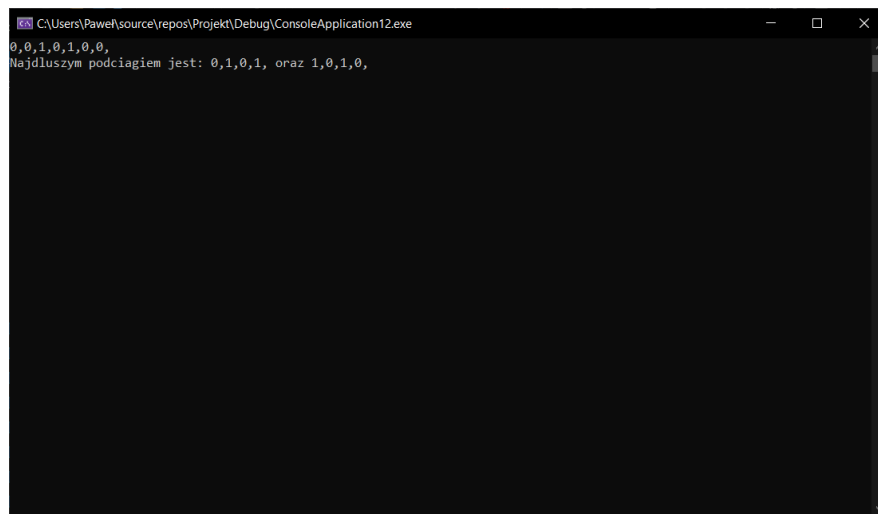
Schemat blokowy



Pseudokod

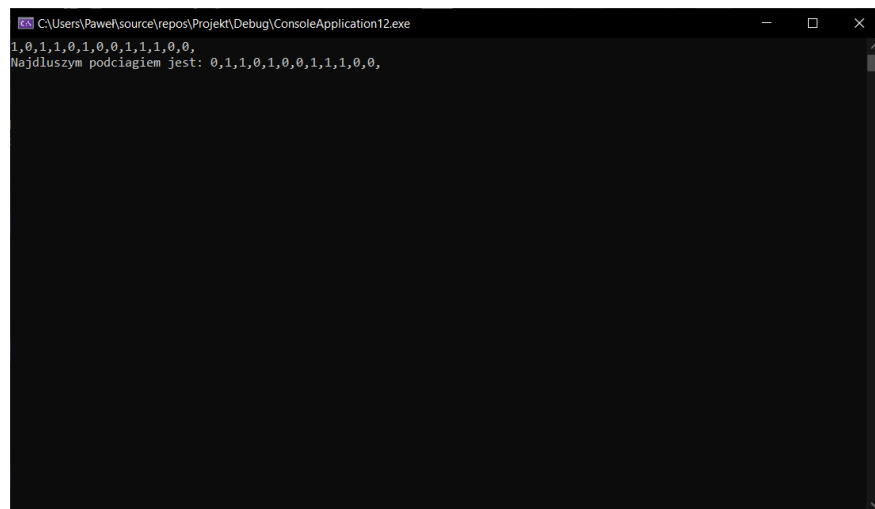
```
Wczytaj tab[];
Dla i=0 do n-1 powtarzaj
    Jeżeli tab[i]==1
        sum=1
    W przeciwnym razie
        sum=-1
    Dla j=i+1 do n powtarzaj
        sum = sum + tab[j]
    Jeżeli sum==0 oraz j-i+1>maks
        start=i
        maks=j-i+1
Zacznij pętle dla i=start do maks-1
    Wypisz tab[i]
Dla i = start do n-1 powtarzaj
    Jeżeli tab[i]==1
        sum2=1
    W przeciwnym razie
        sum2=-1
    Dla j=i+1 do n powtarzaj
        sum2 = sum2 + tab[j]
        Jeżeli sum2==0 oraz j-i+1>maks
            start2=i
            maks2=j-i+1
Jeżeli maks2 == maks
    Zacznij pętle dla i=start2 do maks2-1
        Wypisz tab[i]
```

Działanie kodu



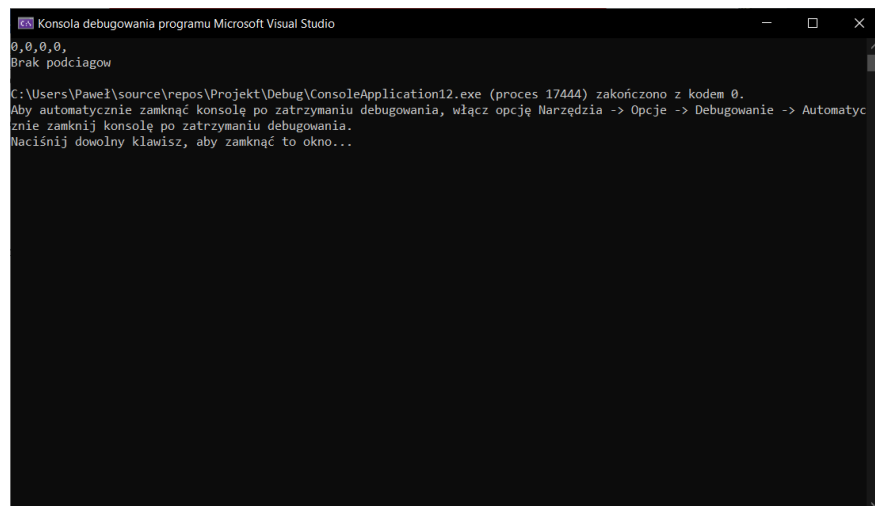
```
C:\Users\Paweł\source\repos\Projekt\Debug\ConsoleApplication12.exe
0,0,1,0,1,0,0,
Najdłuższym podciągiem jest: 0,1,0,1, oraz 1,0,1,0,
```

Rysunek 1: Dwa podciągi



```
C:\Users\Paweł\source\repos\Projekt\Debug\ConsoleApplication12.exe
1,0,1,1,0,1,0,0,1,1,1,0,0,
Najdłuższym podciągiem jest: 0,1,1,0,1,0,0,1,1,1,0,0,
```

Rysunek 2: Jeden podciąg



```
Konsola debugowania programu Microsoft Visual Studio
0,0,0,0,
Brak podciągów

C:\Users\Paweł\source\repos\Projekt\Debug\ConsoleApplication12.exe (proces 17444) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję Narzędzia -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```

Rysunek 3: Brak podciągów

Szkielet kodu

```
#include <bits/stdc++.h>
#include <fstream>

using namespace std;

/*Dla ciągu (w postaci tablicy) zawierającego wyłącznie wartości 0 lub 1, znajdź
podciąg zawierający równą liczbę zer i jedynek, którego długość jest największa
Przykład.
    Wejście: 0,0,1,0,1,0,0
    Wyjście: Najdłuższym podciągiem są: 0,1,0,1 oraz 1,0,1,0
*/

int podciag (int tab[], int n)
{
    int sum = 0; // W tej zmiennej będzie przechowywana suma ciągu
    int maks = -1; // W tej zmiennej będzie przechowywany maksymalny rozmiar podciągu
    int start; // Początek podciągu
    int sum2 = 0; // W tej zmiennej będzie przechowywana suma drugiego ciągu
    int maks2 = -1; // W tej zmiennej będzie przechowywany maksymalny rozmiar drugiego
    podciągu
    int start2; // Początek drugiego podciągu

    // Sprawdzamy po kolei każdy wyraz w tablicy
    for (int i = 0; i < n - 1; i++) {
        if (tab[i] == 0)
            sum = -1; // Jeśli wyraz jest równy 0 to nadajemy zmiennej sum wartość -1
        else
            sum = 1; // Natomiast jeśli ma wartość 1 to nadajemy zmiennej sum wartość
1

        // Liczymy od i-tego wyrazu ciągu różnice lub sume aż do ostatniego wyrazu
        for (int j = i + 1; j < n; j++) {
            if (tab[j] == 0) {
                sum = sum - 1;
            }
            else {
                sum = sum + 1;
            }
            /* Jeśli zmienna sum po dodawaniu i odejmowaniu kolejnych wartości będzie
            równa 0
            to przerywamy pętlę i wykonujemy instrukcje warunkową */
            if (sum == 0 && maks < j - i + 1) {
                maks = j - i + 1; // Do zmiennej maks zapisujemy długość ciągu
                start = i; // Do zmiennej start zapisujemy początek podciągu
            }
        }
    }

    // Jeśli w ciągu nie znajdziemy zmiennej sum równej 0 to oznacza że nie ma w
    naszym ciągu,
    // podciągu o równej liczbie zer i jedynek
    if (maks == -1)
        cout << "Brak podciągów";
        cin.get();
        return 0;

    // W przeciwnym razie wypisujemy nasz podciąg zaczynając od zmiennej zacznij,
    a kończąc na zmiennej suma,
    // której przypisujemy wartość sumy zacznij i maks oraz różnicy 1
    else {
        cout << "Najdluszym podciągiem jest: ";
        int suma = start + maks - 1;
    }
}
```

```

        for (int k = start; k <= suma; k++) {
            cout << tab[k] << ",";
        }
    }
    //Program wykonuje kolejny objęg po ciągu tym razem zaczynając od kolejnego
    wyrazu, znalezione podciągu
    for (int s = start + 1 ; s < n - 1; s++) {
        if (tab[s] == 0)
            sum2 = -1;
        else
            sum2 = 1;

        for (int m = s + 1; m < n; m++) {
            if (tab[m] == 0) {
                sum2 = sum2 - 1;
            }
            else {
                sum2 = sum2 + 1;
            }

            if (sum2 == 0 && maks2 < m - s + 1) {
                maks2 = m - s + 1;
                start2 = s;
            }
        }
    }
    // Teraz sprawdzamy czy drugi podciąg ma taką samą długość.
    // Jeśli tak to wypisujemy go na tych samych warunkach co podciąg pierwszy
    if (maks2 == maks) {
        cout << " oraz ";
        if (maks2 == -1) {

        }
        else
        {
            int suma2 = start2 + maks2 - 1;
            for (int i = start2; i <= suma2; i++) {
                cout << tab[i] << ",";
            }
        }
    }

    return maks, maks2;
}

int main()
{
    int tab[] = {0,0,1,0,1,0,0};
    int size = sizeof(tab) / sizeof(tab[0]);
    podciag(tab, size);
    return 0;
}

```

