

Algorytmy i struktury danych

Projekt

P01 Kamil Bieniek

Dla zadanego ciągu zer i jedynek, znajdź wszystkie dłuższe niż dwuelementowe podciągi, w których zera poprzedza ta sama liczba jedynek.

Przykład:

Wejście [0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0]

Wyjście [1, 1, 1, 0, 0, 0] , [1, 1, 0, 0]

Wejście [0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1]

Wyjście Brak elementów spełniających zadane kryteria.

## Funkcje użyte w programie

1. wpisywanie ();

-Wpisuje ciąg losowych liczby z zakresu 0 – 1 do pliku in.txt

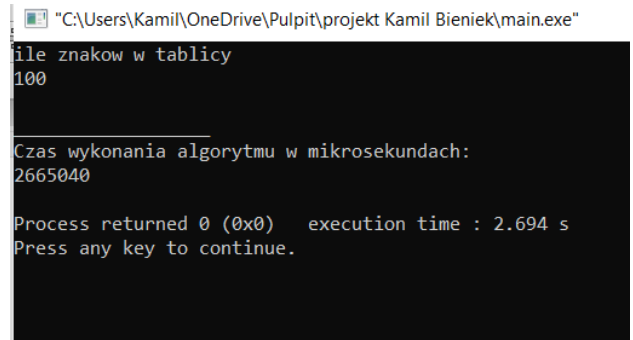
2. kod(z);

- wyciąga z pliku in.txt podciągi spełniające kryteria i wpisuje je do pliku out.txt

Opis działania programu Program ma za zadanie wpisanie do pliku in.txt losowych wygenerowanych liczb z zakresu 0 – 1 w ilości podanej przez użytkownika. Następnie funkcja kod(z) wyciąga z pliku podciągi zer i jedynek w których liczba zer jest taka sama jak liczba jedynek poprzedzająca je a następnie wpisuje je do pliku w nawiasach kwadratowych. W programie jest też zawarty kod zliczający nam czas wykonywanie całego kodu w mikro sekundach.

## Screeny z działania programu

Screen konsoli:

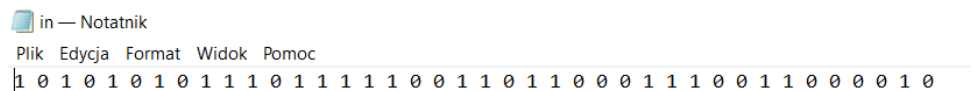


```
"C:\Users\Kamil\OneDrive\Pulpit\projekt Kamil Bieniek\main.exe"
ile znakow w tablicy
100

Czas wykonania algorytmu w mikrosekundach:
2665040

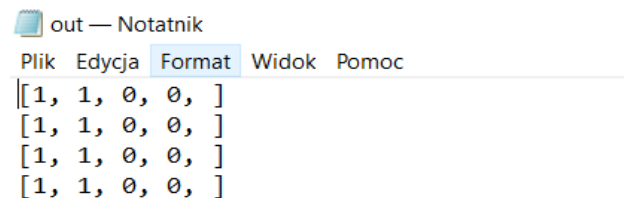
Process returned 0 (0x0)   execution time : 2.694 s
Press any key to continue.
```

Screen z pliku in:



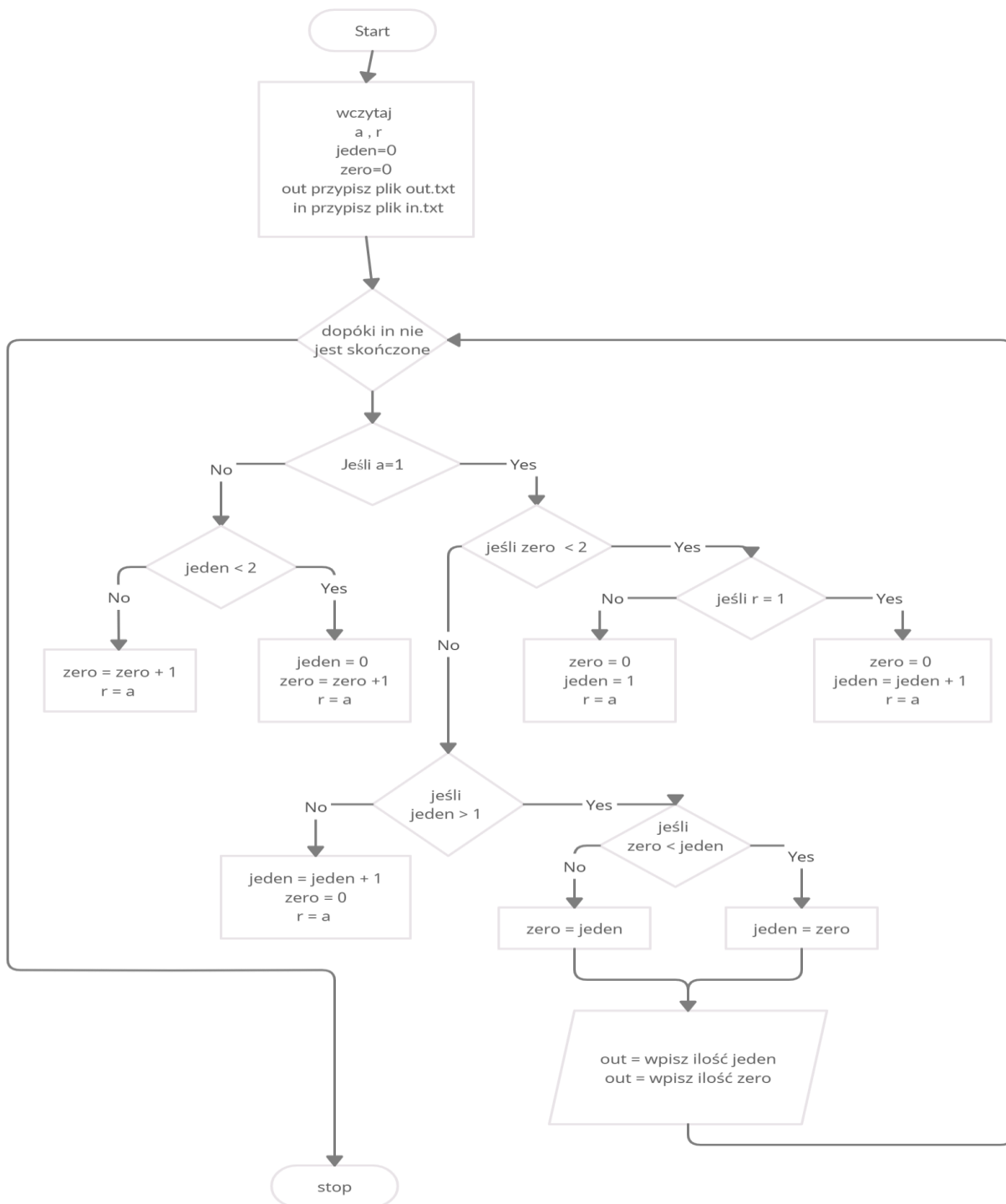
```
in — Notatnik
Plik  Edycja  Format  Widok  Pomoc
1 0 1 0 1 0 1 0 1 1 1 0 1 1 1 1 0 0 1 1 0 1 1 0 0 0 1 1 1 0 0 1 1 0 0 0 0 1 0
```

Screen z pliku out:



```
out — Notatnik
Plik  Edycja  Format  Widok  Pomoc
[1, 1, 0, 0, ]
[1, 1, 0, 0, ]
[1, 1, 0, 0, ]
[1, 1, 0, 0, ]
```

# Schemat Blokowy



## Pseudokod

```
kod(z);

in= dane z pliku in.txt

dopóki (plik nie jest skończony){
  liczba z pliku >>a;
  jeżeli(liczba z pliku = 1){
    jeżeli(licznik zer < 2)
    {
      jeżeli(ostatnia liczba = 1) {
        Licznik zer = 0; licznik jedynek + 1 ;ostatnia liczba=a;}
      W przeciwnym wypadku{
        licznik zer=0;licznik jedynek = 1;ostatnia liczba=a;}
      }
    w przeciwnym wypadku {
      jeżeli (licznik jedynek >1){
        jeżeli(licznik zer > licznik jedynek){
          {licznik jedynek = licznik zer;}
          w przeciwnym wypadku
          {licznik zer = licznik jedynek;}
        }
      }
    }

    dla(i=0;i < licznik jedynek;i++ )
    {podciąg =podciąg +"1,"}
    dla(i=0;i<licznik zer;i++)
    {podciąg = podciąg+"0,"}"
    Do pliku wpisz podciąg
    Licznik jedynek =0 ;licznik zer=0 licznik jedynek++; ostatnia liczba =a
    w przeciwnym wypadku{
      Jeżeli (licznik jedynek <2){
        Licznik jedynek =0;licznik zer++; ostatnia liczba = a}
      w przeciwnym wypadku{
        Licznik zer ++;ostatnia liczba =a}}
    Jeżeli (licznik zer –1>1 i licznik jedynek >1){
      Jeżeli (licznik zer > licznik jedynek ){
        licznik jedynek = licznik zer}
      w przeciwnym wypadku{
        Licznik zer = licznik jedynek}
    }
    dla(i=0;i< licznik jedynek ; i++)
    Podciąg=podciąg +"1 "}
    dla(i=0;i<licznik zer; i++)
    Podciąg= podciąg + "0 "}
    Do pliku wpisz podciąg
```

## Czas działanie algorytmu w mikrosekundach

|                   |                     |
|-------------------|---------------------|
| Dla 10 elementów  | 577866 mikrosekund  |
| Dla 100 elementów | 792387 mikrosekund  |
| Dla 300elementów  | 836799 mikrosekund  |
| Dla 600elementów  | 1035572 mikrosekund |
| Dla 900elementów  | 1152487 mikrosekund |
| Dla 1200elementów | 1232008 mikrosekund |
| Dla 1500elementów | 1535651 mikrosekund |



