P O L I T E C H N I K A R Z E S Z O W S K A
im. Ignacego Łukasiewicza
WYDZIAŁ MATEMATYKI I FIZYKI STOSOWANEJ

JAKUB BARAN

Projekt C++

Zadanie 2
kierunek studiów: Inżynieria i Analiza Danych

Opiekun pracy:

Mariusz Borkowski

Rzeszów 2022

Sortowanie bąbelkowe

Sortowanie bąbelkowe (ang. bubble sort) – prosta metoda sortowania o złożoności obliczeniowejO(n^2) i pamięciowej O(n). Polega na porównywaniu dwóch kolejnych elementów i zamianie ich kolejności, jeżeli zaburza ona porządek, w jakim się sortuje tablicę. Sortowanie kończy się, gdy podczas kolejnego przejścia nie dokonano żadnej zmiany.

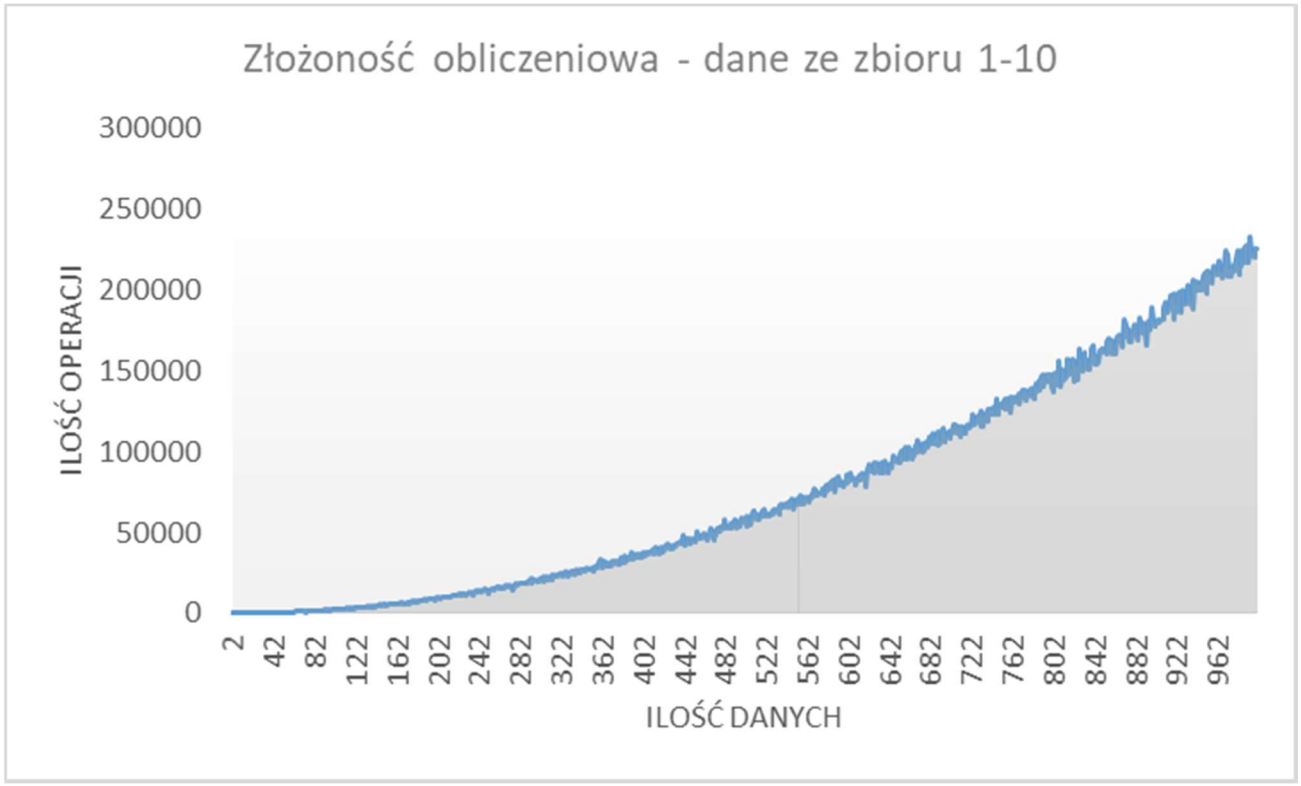
```
zmienna N = 20
tablica d[N]
zmienne i,s,smin,smax
dla i = 0 do N-1
    d[i] = liczba pseudolosowa z zakresu 0 do 99
smin = 0, smax = N – 1
zrób:
    p=-1
    dla i = smin do smax
        jeśli(d[i] > d[i+1])
            zamień(d[i], d[i+1])
            jeśli s<0 to smin = i
            s=i
    smin = smin-1
    smax = s
dopóki p >= 0 powtarzaj
```

Sortowanie gnoma

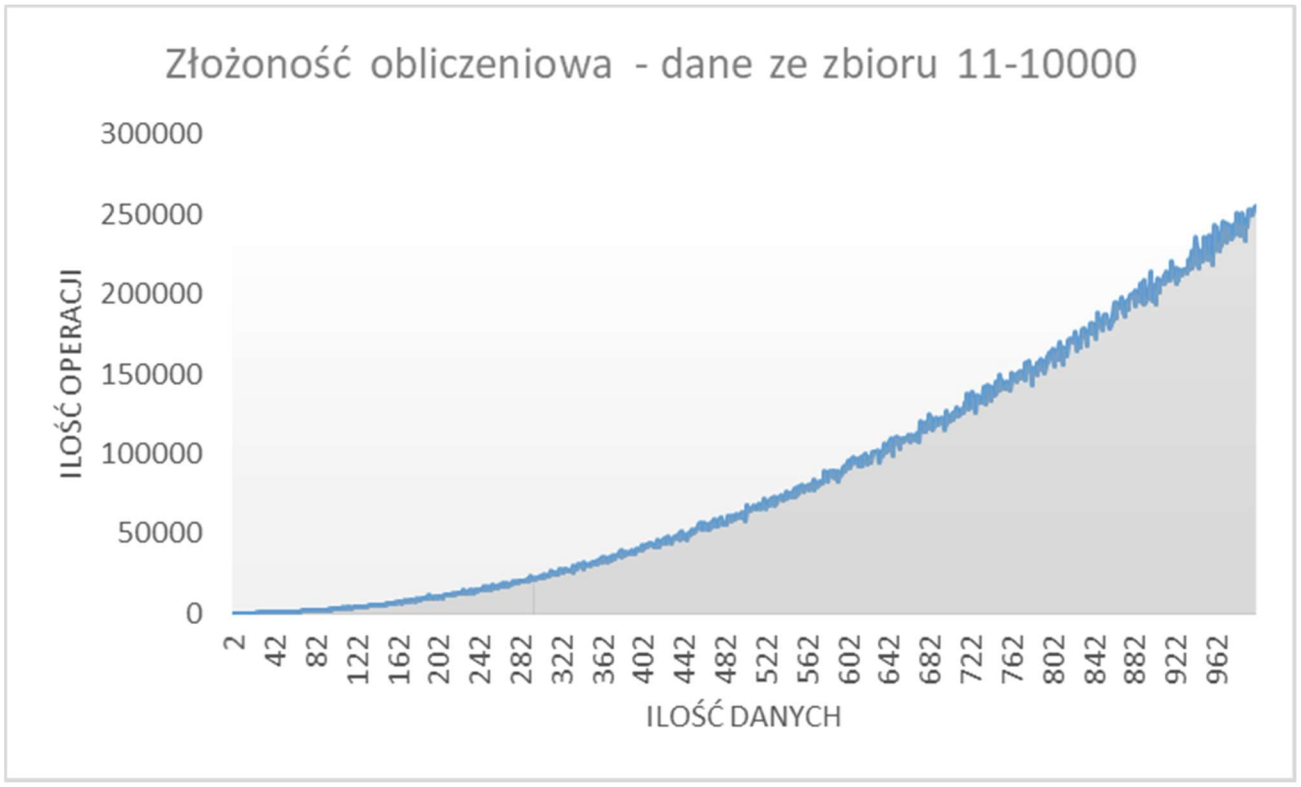
Sortowanie gnoma to algorytm sortowania, który porównuje każdy element z jego następnikiem i przesuwa elementy nieodpowiedniej kolejności w kierunku końca listy. W ten sposób, każde przejście powoduje przesunięcie jednego elementu na swoje miejsce końcowe. Jest to bardziej skuteczne niż sortowanie bąbelkowe, ale nadal jest to algorytm złożoności O(n^2).

```
zmienna N = 20
tablica d[N]
dla i = 0 do N-1
    d[i] = liczba pseudolosowa z zakresu 0 do 99
zmienna pos = 0
zmienna ost_pos = 0
dopóki pos <  N to wykonaj:
    jeśli pos == ost_pos lub d[pos] >= d[pos-1]
        pos = ost_pos + 1
        ost_pos = pos;
    inaczej
        zamień(d[pos], d[pos-1])
        pos = pos – 1
```

Sortowanie bąbelkowe złożoność obliczeniowa algorytmu wybierając tylko dane ze zbioru 1-10 do posortowania

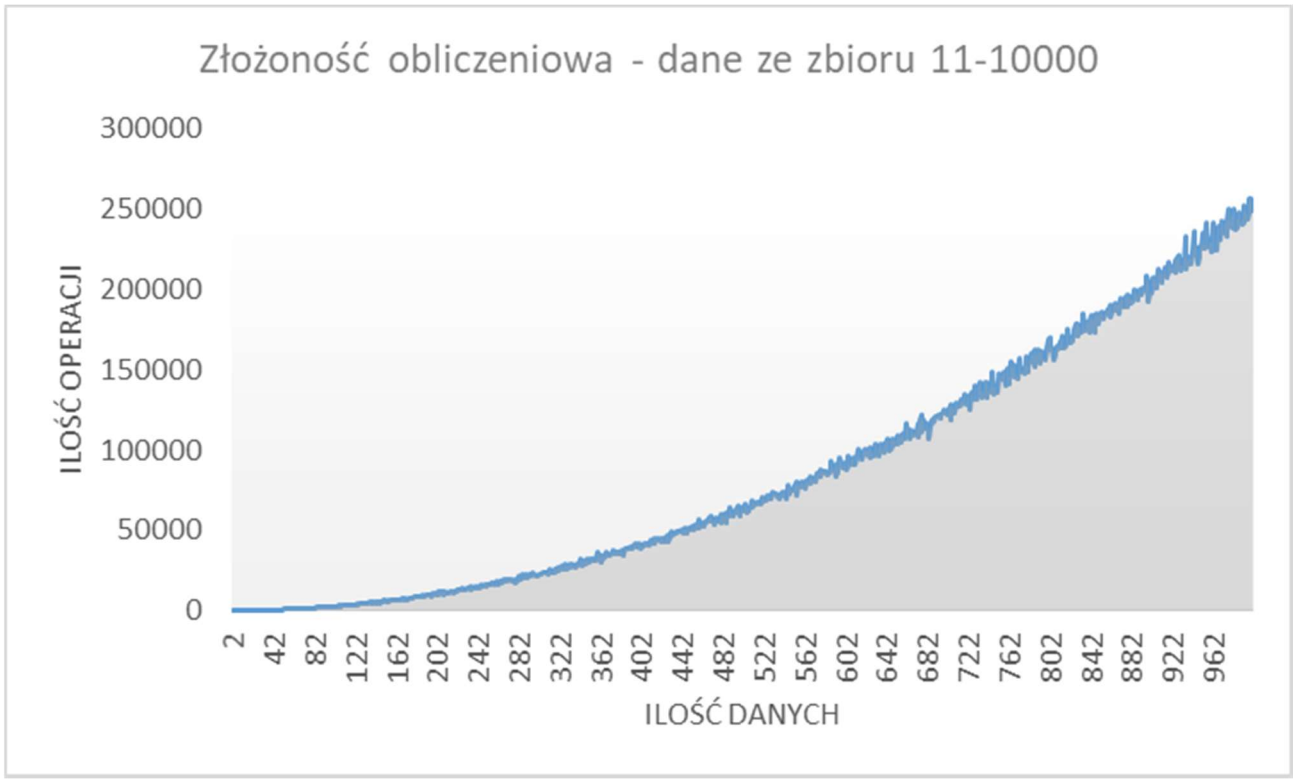


Sortowanie bąbelkowe złożoność obliczeniowa algorytmu wybierając tylko dane ze zbioru 11-10000 do posortowania

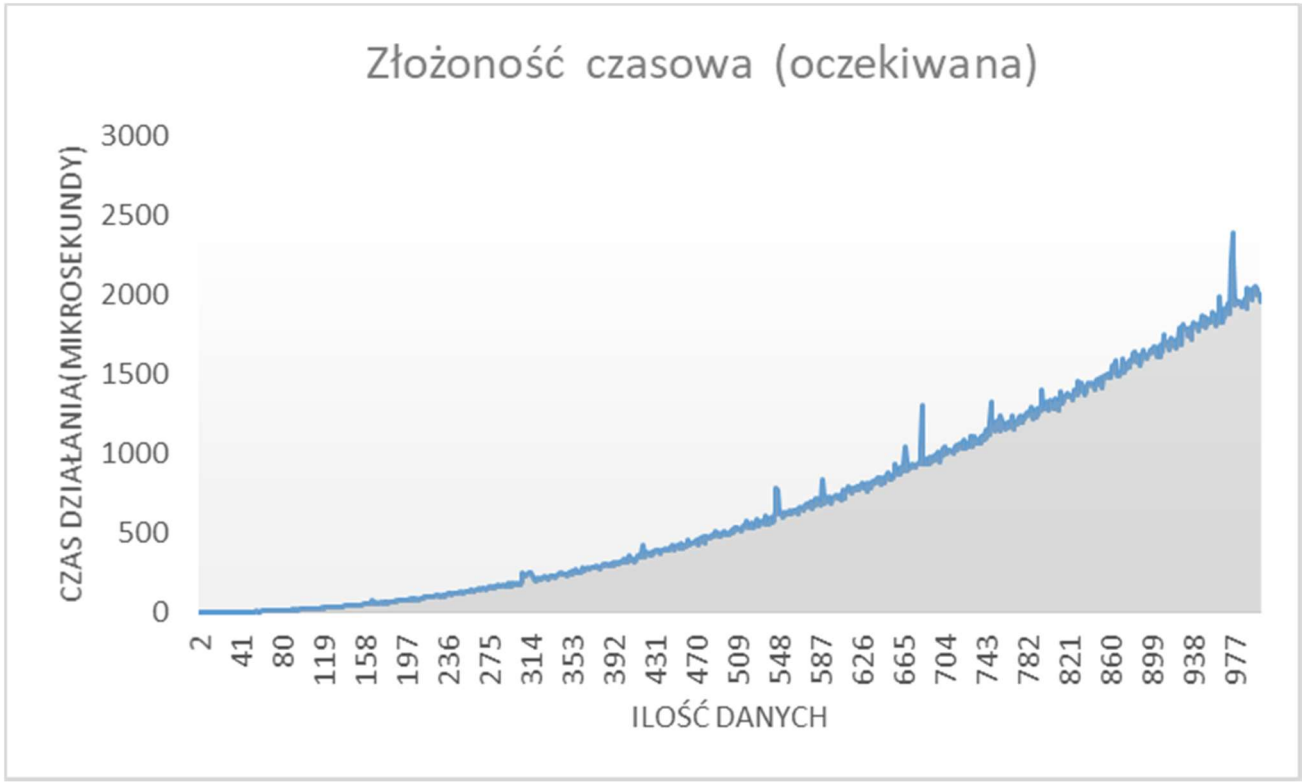


Sortowanie gnoma złożoność obliczeniowa algorytmu wybierając tylko dane ze zbioru 1-10 do posortowania

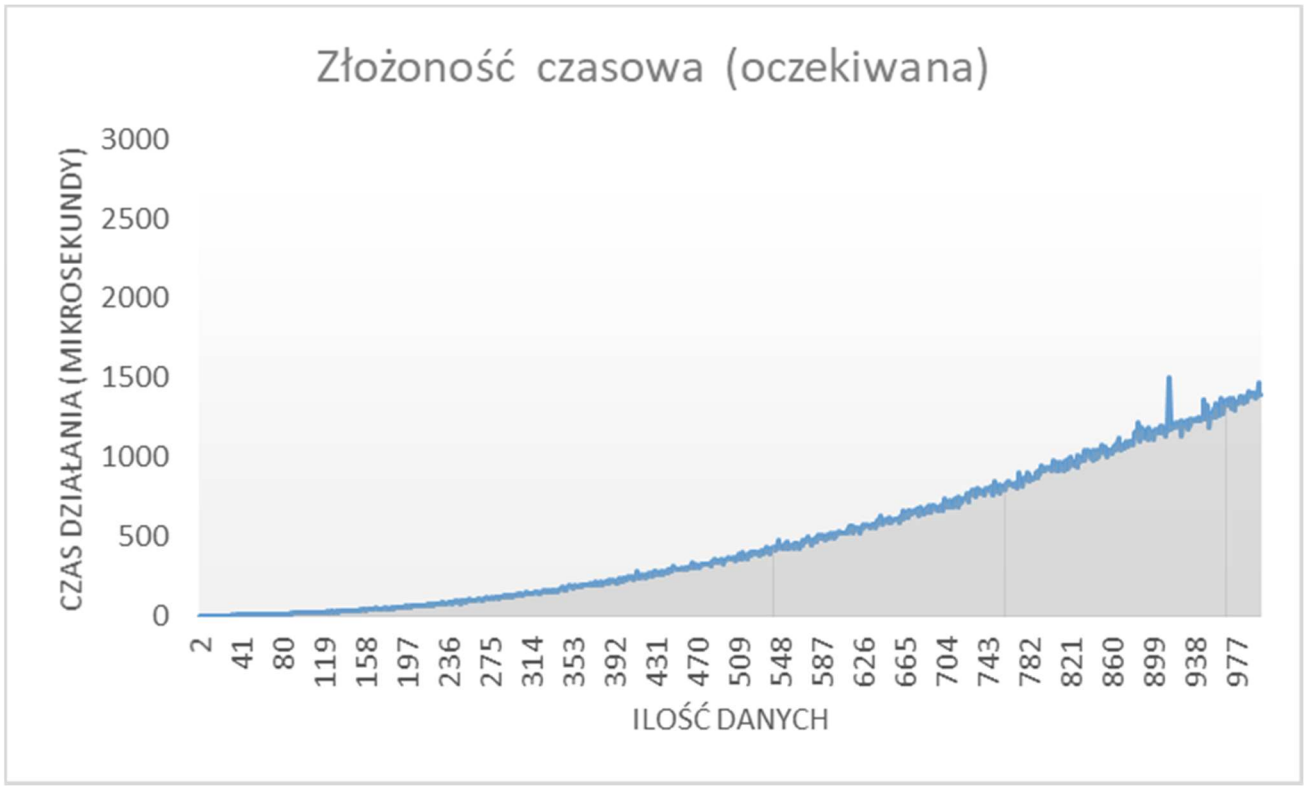
Sortowanie gnoma złożoność obliczeniowa algorytmu wybierając tylko dane ze zbioru 11-10000 do posortowania



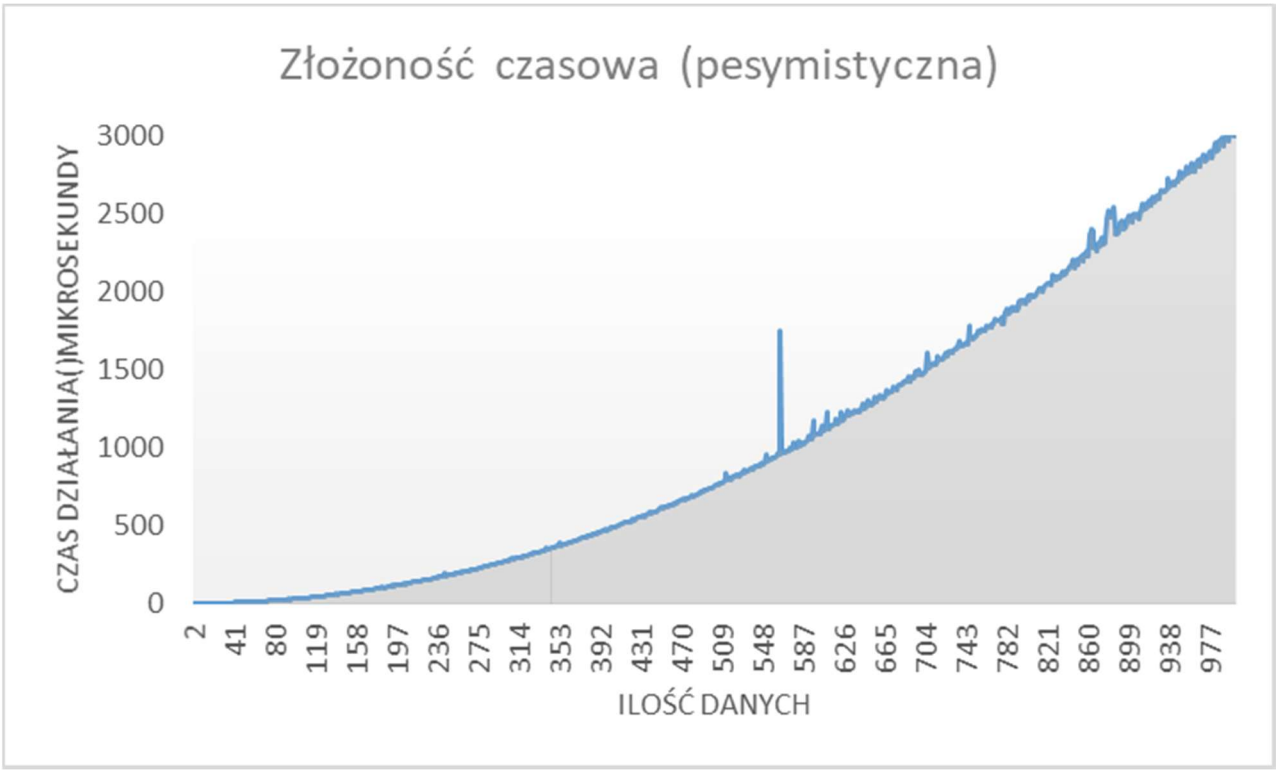
Złożoność czasowa algorytmu (oczekiwana) sortowania bąbelkowego w skali do 3000 ms



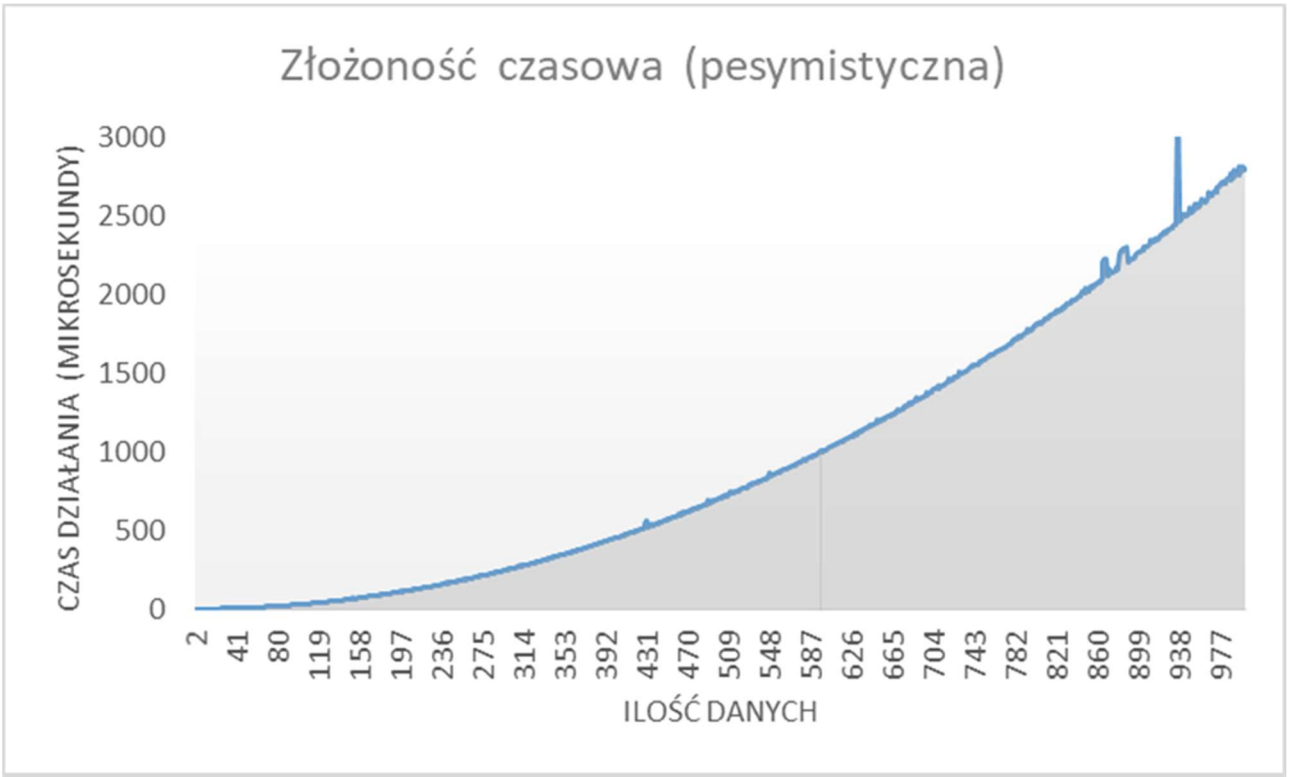
Złożoność czasowa (oczekiwana) sortowania gnoma



Złożoność czasowa algorytmu (pesymistyczna) sortowania bąbelkowego

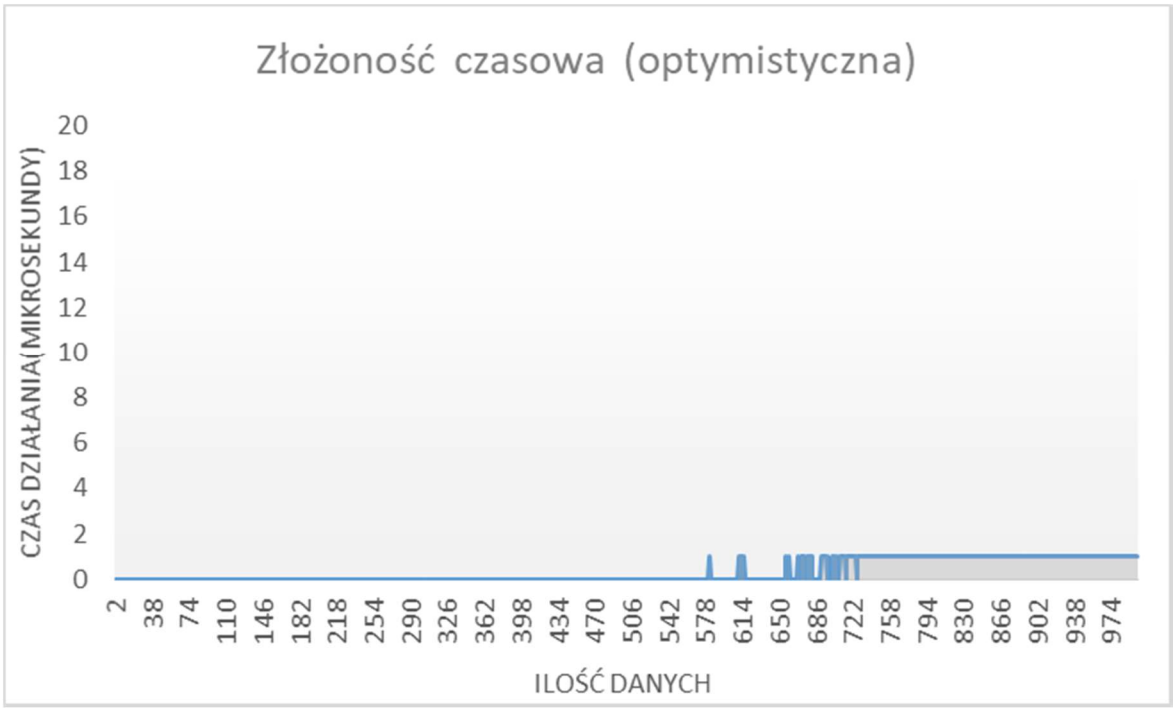


Złożoność czasowa algorytmu (pesymistyczna) sortowania gnoma

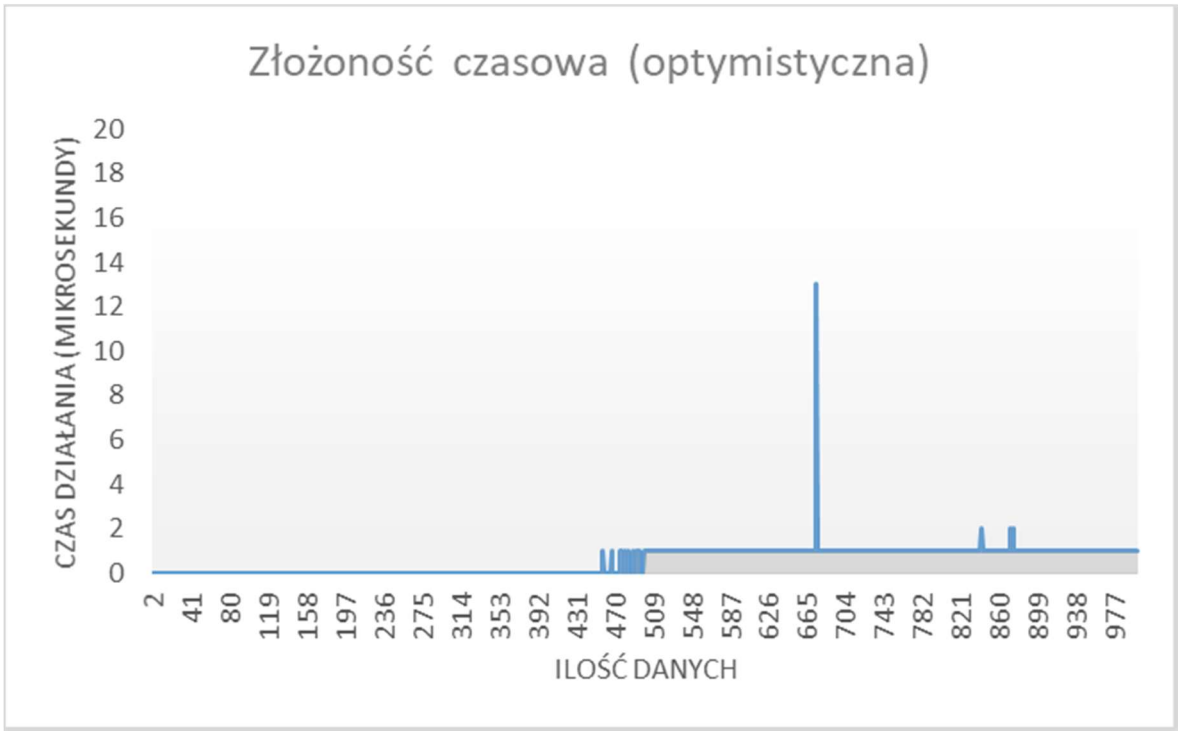


Złożoność czasowa algorytmu(pozytywna) sortowania przez scalanie

Złożoność czasowa algorytmu (optymistyczna) sortowania bąbelkowe. Dlatego, że czas działania jest dużo krótszy musiałem zmienić skale do 20 ms



Złożoność czasowa algorytmu (optymistyczna) sortowania bąbelkowe



Porównanie

Sortowanie gnoma jest algorytmem sortowania, który polega na przesuwaniu elementów w tablicy tak, aby elementy mniejsze od danego elementu (zwanego gnomem) znalazły się po jego lewej stronie, a elementy większe po jego prawej stronie. Sortowanie to jest dość podobne do sortowania bąbelkowego, jednak różni się tym, że sortowanie bąbelkowe przesuwa elementy tak, aby największe elementy znalazły się na końcu tablicy, podczas gdy sortowanie gnoma przesuwa elementy tak, aby elementy były uporządkowane od najmniejszego do największego.

Algorytm sortowania bąbelkowego polega na porównywaniu każdego elementu z jego bezpośrednim sąsiadem i zamianie ich miejscami, jeśli są one w złej kolejności. Proces ten jest powtarzany wiele razy, aż wszystkie elementy znajdą się w odpowiedniej kolejności.

W skrócie: Sortowanie gnoma jest algorytmem polegającym na porządkowaniu elementów od najmniejszego do największego przez przesuwanie ich po tablicy, a sortowanie bąbelkowe polega na porządkowaniu elementów przez porównywanie ich z bezpośrednim sąsiadem i zamianę miejscami, jeśli są one w złej kolejności.

Wnioski po porównaniu złożoności wykresów

W skrócie liczby operacji wykonywanych w obu algorytmach są bardzo podobne wynika to z tego, że ich złożoności są takie same – obie wynoszą O(n^2), i również lekko się zwiększają jeśli poszerzymy zbiór losowanych danych. W przypadku złożoności czasowej, widać że przy losowych danych lepiej wypada sortowanie gnoma – wykonanie programu zajmuje mniej czasu. Jeśli chodzi o przypadki pesymistyczny czyli przy odwrotnie dobieranych danych i optymistyczny czyli praktycznie już posortowanych danych są one bardzo podobne. Czyli sortowanie gnoma jest efektywniejsze przynajmniej jeśli chodzi o przypadek oczekiwany.