

Imię i Nazwisko	Przedmiot	Data oddania	Ocena
Paulina Roszkowska Aleksandra Szum	Algorytmy i struktury danych	31 stycznia 2020	
Sprawozdanie IV			
Temat sprawozdania		Sortowanie	

1 Opis ćwiczenia

Podczas zajęć otrzymano listę liczb od Prowadzącego zajęcia. Celem ćwiczenia było wykonanie sortowania bąbelkowego. Zadanie polegało na porównaniu dwóch kolejnych elementów i zamianie ich kolejności, tak aby liczby były uporządkowane rosnąco. Po 9-tej iteracji uzyskano zestaw liczb posortowanych rosnąco. Po zakończeniu sortowania wyznaczono empiryczny koszt algorytmu i porównano go z kosztem teoretycznym.

2 Wstęp teoretyczny

Sortowanie bąbelkowe 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. Algorytm opiera się na zasadzie maksimum, to znaczy, że każda liczba jest mniejsza lub równa od liczby maksymalnej. Porównując kolejno liczby można wyznaczyć największą z nich. Następnie ciąg częściowo posortowany (mający liczbę maksymalną), można skrócić o tę liczbę i ponowić szukanie maksimum, już bez elementów odrzuconych i tak dugo, aż zostanie nam jeden element. Otrzymane kolejne maksima są coraz mniejsze przez co ciąg jest uporządkowany. Sortowanie bąbelkowe zobrazowano na filmie: <https://www.youtube.com/watch?v=lyZQPjUT5B4>

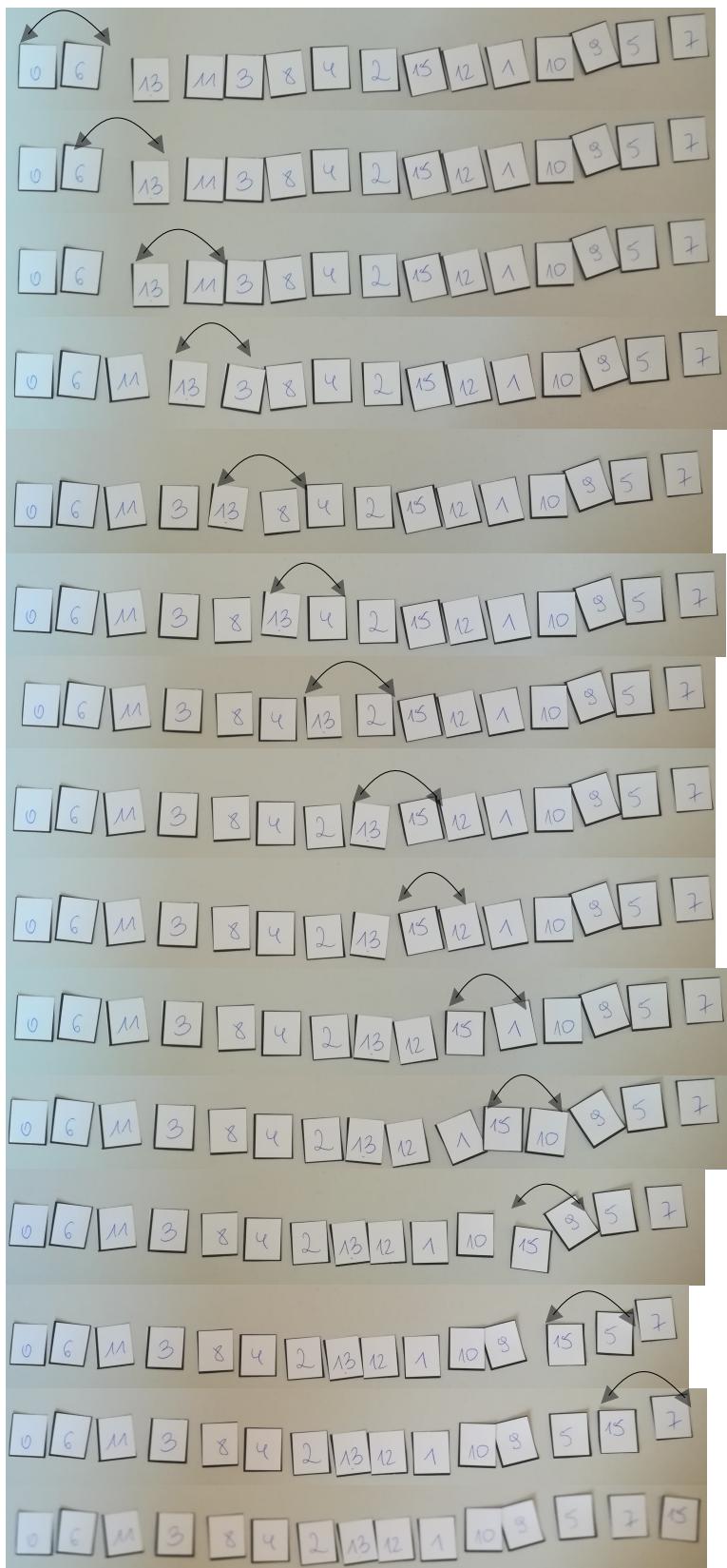
Koszt teoretyczny algorytmu sortowania bąbelkowego wynosi $O(n)$, natomiast złożoność średnia $O(n^2)$.

Empirycznym kosztem operacji nazywa się koszt złożoności obliczeniowej uzyskanej podczas analizy empirycznej. Analiza empiryczna polega na przeprowadzeniu symulacji i określonych obliczeń. Aby dokonać analizy empirycznej, należy dysponować poprawną implementacją algorytmu oraz dany zestaw danych wejściowych.

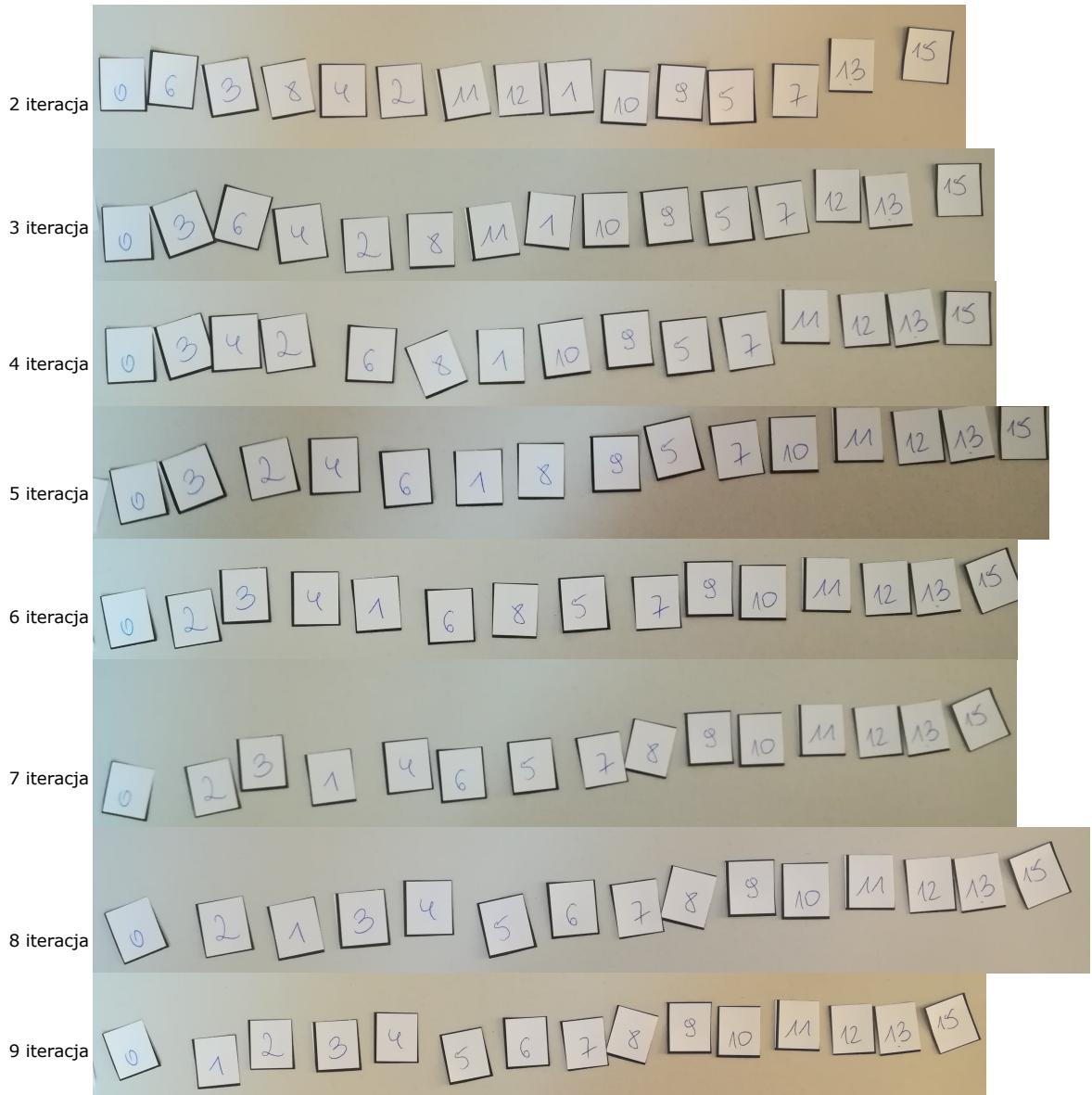
3 Obliczenia numeryczne

Kod znajduje się na stronie https://github.com/aleksandraszum/AiSD_projekty/tree/master/report4.

4 Analiza wyników



Rysunek 1: Efekty działania sortowania bąbelkowego w pierwszej iteracji.



Rysunek 2: Efekt działania sortowania bąbelkowego w iteracji od 2 do 9.

Powyżej znajdują się zdjęcia przedstawiające sortowanie bąbelkowe zadanej tablicy. Rysunek 1 pokazuje wykonywanie się algorytmu krok po kroku w pierwszej iteracji, natomiast rysunek 2 pokazują efekt końcowy działania algorytmu w danej iteracji.

Złożoność minimalna algorytmu sortowania bąbelkowego wynosi $O(n)$, natomiast złożoność średnia $O(n^2)$, gdzie n jest liczbą elementów. W przypadku analizowanej tablicy wartość n wynosi $n = 15$.

Opracowany przez Autorki algorytm sortowania bąbelkowego dokonał posortowania tablicy w 9-ciu iteracjach. 10 iteracja algorytmu służy po to, aby sprawdzić, czy wszystkie liczby znajdują się w odpowiedniej kolejności.

Za empiryczny koszt algorytmu uznano liczbę wywołań pętli `for`, której zadaniem jest iteracja po elementach tablicy oraz liczbę wywołań pętli `while`. W przypadku listy liczb otrzymanych od Prowadzącego zajęcia, koszt wyniósł 150.

W celu wyznaczenia wzoru dotyczącego kosztu empirycznego, przeprowadzono także posortowanie posortowanej już listy liczb oraz posortowanie malejąco posortowanej tablicy. Wyniki znajdują się w tabeli 1.

Tabela 1: Zebrane wyniki dotyczące empirycznego kosztu operacji sortowania bąbelkowego

Empiryczny koszt operacji sortowania bąbelkowego		
Sortowanie listy liczb podanej przez Prowadzącego	Sortowanie rosnące listy liczb posortowanej rosnąco	Sortowanie malejące listy liczb posortowanej rosnąco
140	14	210

Od Prowadzącego otrzymano 15 liczb. Pętla `for` iterowała od zerowego do przedostatniego elementu listy. W przypadku najbardziej optymistycznym (sortowanie bąbelkowe listy liczb posortowanej), empiryczna złożoność wynosi $O(n)$, gdzie n jest liczbą elementów tablicy, a w przypadku analizowanej listy wynosi $n = 15$. W przypadku najbardziej pesymistycznym, czyli sortowanie listy posortowanej w odwrotnej kolejności, złożoność obliczeniowa jest największa i wyniosła $O(225)$. Empiryczna złożoność sortowania listy liczb podanej przez Prowadzącego wyniosła $O(150)$. Zauważono, że liczba 250 jest wynikiem działania mnożenia $15 \cdot 15$, natomiast liczba 150 można zapisać jako $150 = 15 \cdot 10$. W związku z tym wyznaczono empiryczną złożoność, którą można zapisać wzorem: $O(c \cdot n)$, gdzie n to liczba elementów w tabeli, a c to liczba przeprowadzonych iteracji.

5 Wnioski

Udowodniono, że złożoność minimalna algorytmu sortowania bąbelkowego wynosi $O(n)$. Taką złożoność otrzymuje się w przypadku sortowania bąbelkowego posortowanej już tablicy liczb.

Przy użyciu danych pesymistycznych, czyli tablicy liczb posortowanych w odwrotnej kolejności, koszt obliczeniowy algorytmu jest największy i wynosi $O(n^2)$.

Empiryczny koszt algorytmu można przedstawić zależnością $O(c \cdot n)$, gdzie $1 \leq c \leq$.

Do zalet sortowania bąbelkowego można zaliczyć prostotę implementacji.

Do wad sortowania bąbelkowego można zaliczyć dużą złożoność obliczeniową (szczególnie w przypadku sortowania ogromnej ilości danych).

6 Bibliografia

1. Wikipedia, *Sortowanie bąbelkowe*, dostęp online:
https://pl.wikipedia.org/wiki/Sortowanie_b%C4%85belkowe
2. ANALIZA SPRAWNOŚCI ALGORYTMÓW , dostęp online: https://xion.org.pl/files/texts/mgt/html/M_B.html
3. Andrzej Kapanowski, *Analiza algorytmów. Wprowadzenie*, dostęp online: <http://users.uj.edu.pl/~ufkapano/algorytmy/lekcia08/analiza.html>