**SPRAWOZDANIE**

**PROEJKT Z PRZEDMIOTU „ALGORYTMY I STRUKTURY DANYCH”**

**TEMAT**

„Zaimplementuj sortowanie przez wstawianie oraz sortowanie kopcowe.”

Tomasz Kozioł, PRz, semestr 2020/2021, Inżynieria i Analiza Danych

Spis treści

Spis treści

[**1.** **Podstawy teoretyczne algorytmów sortowania** 3](#_Toc61278496)

[1.1 Sortowanie przez kopcowanie 3](#_Toc61278497)

[1.1.1 Opis algorytmu 3](#_Toc61278498)

[1.1.2 Kod źródłowy algorytmu sortowania przez kopcowanie: 4](#_Toc61278499)

[1.2 Sortowanie przez wstawianie 5](#_Toc61278500)

[1.2.1 Opis algorytmu 5](#_Toc61278501)

[1.2.2 Kod źródłowy algorytmu sortowania przez wstawianie: 6](#_Toc61278502)

[**2.** **Schematy blokowe** 7](#_Toc61278503)

[**3.** **Testy porównujące działanie obu algorytmów** 9](#_Toc61278504)

[**4.** **Pseudokod** 10](#_Toc61278505)

[4.1 Lista kroków dla algorytmu sortowania przez kopcowanie 10](#_Toc61278506)

[4.1.1 Budowa kopca: 10](#_Toc61278507)

[4.1.2 Rozbiór kopca 10](#_Toc61278508)

[4.2 Lista kroków dla algorytmu sortowania przez wstawianie 11](#_Toc61278509)

# **Podstawy teoretyczne algorytmów sortowania**

## Sortowanie przez kopcowanie

Kopiec jest drzewem binarnym, w którym wszystkie węzły spełniają następujący warunek (zwany warunkiem kopca):

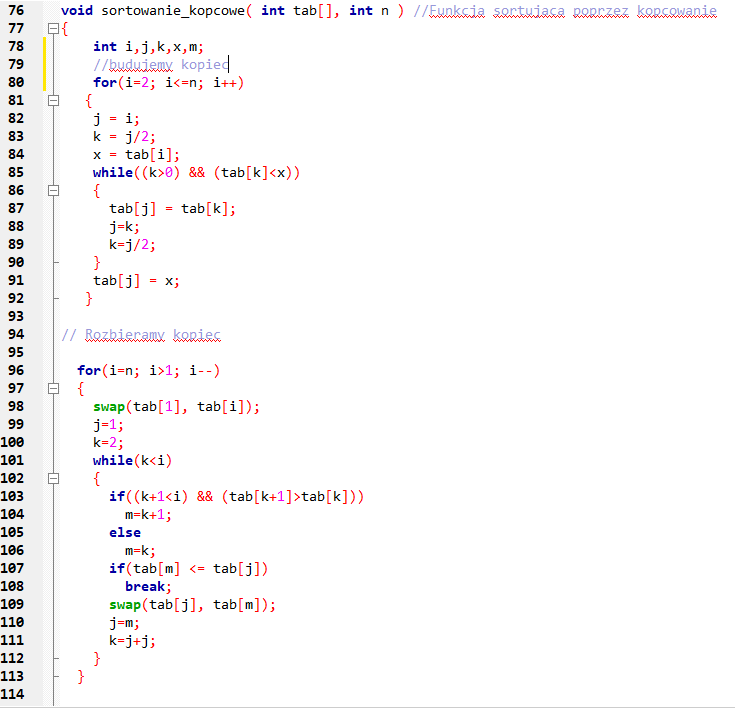
Węzeł nadrzędny jest większy lub równy węzłom potomnym (w porządku malejącym relacja jest odwrotna - mniejszy lub równy).

Konstrukcja kopca jest nieco bardziej skomplikowana od konstrukcji drzewa binarnego, ponieważ musimy dodatkowo troszczyć się o zachowanie warunku kopca. Zatem po każdym dołączeniu do kopca nowego węzła, sprawdzamy odpowiednie warunki i ewentualnie dokonujemy wymian węzłów na ścieżce wiodącej od dodanego węzła do korzenia.

### Opis algorytmu

Podstawą algorytmu jest użycie kolejki priorytetowej zaimplementowanej w postaci binarnego kopca zupełnego. Zasadniczą zaletą kopców jest stały czas dostępu do elementu maksymalnego (lub minimalnego) oraz logarytmiczny czas wstawiania i usuwania elementów; ponadto łatwo można go implementować w postaci tablicy.

### 1.1.2 Kod źródłowy algorytmu sortowania przez kopcowanie:



Algorytm 1 Kopcowanie

Algorytm sortowania przez kopcowanie składa się z dwóch faz. W pierwszej sortowane elementy reorganizowane są w celu utworzenia kopca. W drugiej zaś dokonywane jest właściwe sortowanie.

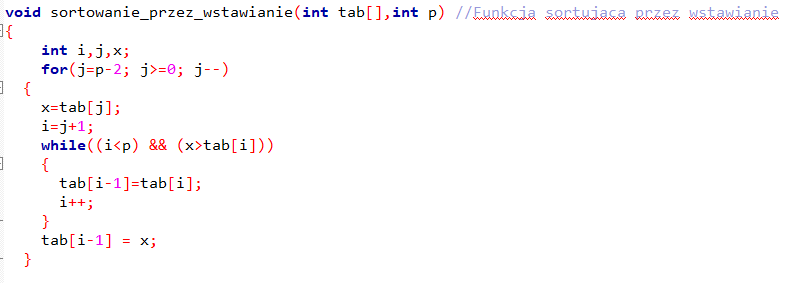
## 1.2 Sortowanie przez wstawianie

Algorytm sortowania przez wstawianie można porównać do sposobu układania kart pobieranych z talii. Najpierw bierzemy pierwszą kartę. Następnie pobieramy kolejne, aż do wyczerpania talii. Każdą pobraną kartę porównujemy z kartami, które już trzymamy w ręce i szukamy dla niej miejsca przed pierwszą kartą starszą (młodszą w przypadku porządku malejącego). Gdy znajdziemy takie miejsce, rozsuwamy karty i nową wstawiamy na przygotowane w ten sposób miejsce (stąd pochodzi nazwa algorytmu - sortowanie przez wstawianie, ang. Insertion Sort). Jeśli nasza karta jest najstarsza (najmłodsza), to umieszczamy ją na samym końcu. Tak porządkujemy karty.

### 1.2.1 Opis algorytmu

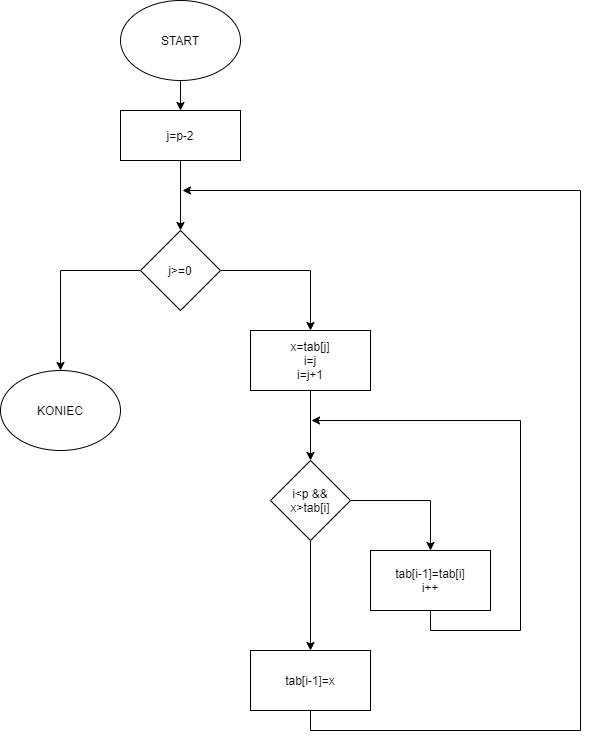
Algorytm sortowania przez wstawianie będzie składał się z dwóch pętli. Pętla główna (zewnętrzna) symuluje pobieranie kart, czyli w tym wypadku elementów zbioru. Odpowiednikiem kart na ręce jest tzw. lista uporządkowana (ang. sorted list), którą sukcesywnie będziemy tworzyli na końcu zbioru (istnieje też odmiana algorytmu umieszczająca listę uporządkowaną na początku zbioru). Pętla sortująca (wewnętrzna) szuka dla pobranego elementu miejsca na liście uporządkowanej. Jeśli takie miejsce zostanie znalezione, to elementy listy są odpowiednio rozsuwane, aby tworzyć miejsce na nowy element i element wybrany przez pętlę główną trafia tam. W ten sposób lista uporządkowana rozrasta się. Jeśli na liście uporządkowanej nie ma elementu większego od wybranego, to element ten trafia na koniec listy. Sortowanie zakończymy, gdy pętla główna wybierze wszystkie elementy zbioru.

### 1.2.2 Kod źródłowy algorytmu sortowania przez wstawianie:

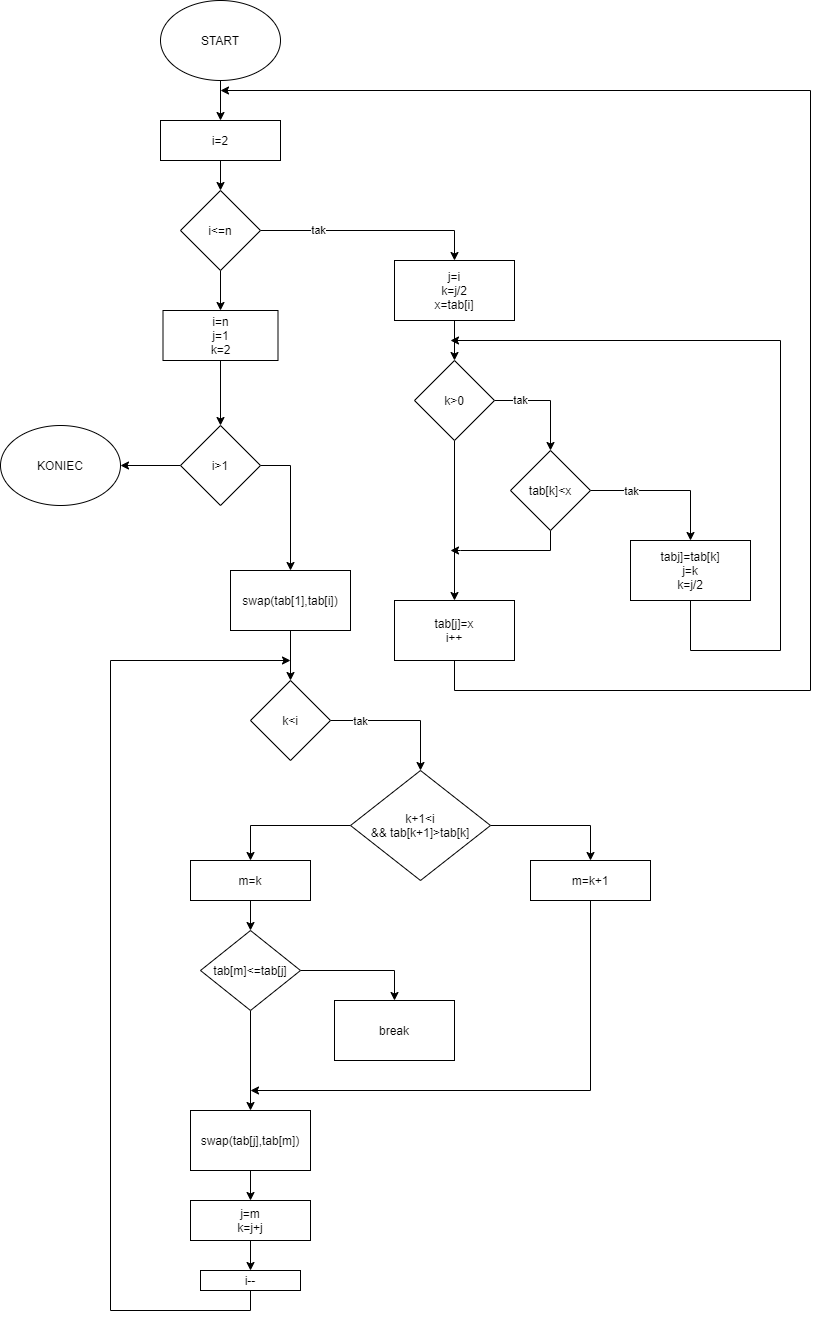


Algorytm 2 Wstawianie

# **Schematy blokowe**



Schemat blokowy 1 Sortowanie przez wstawianie



Schemat blokowy 2 Sortowanie przez kopcowanie

# **Testy porównujące działanie obu algorytmów**

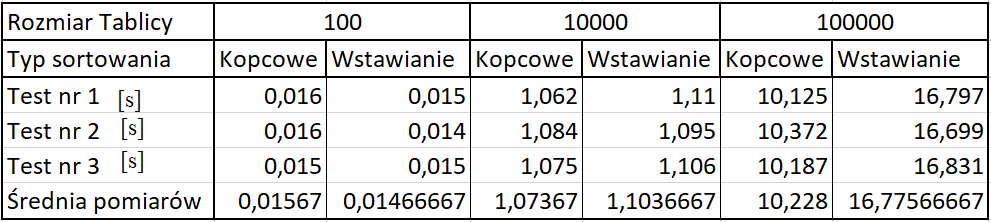
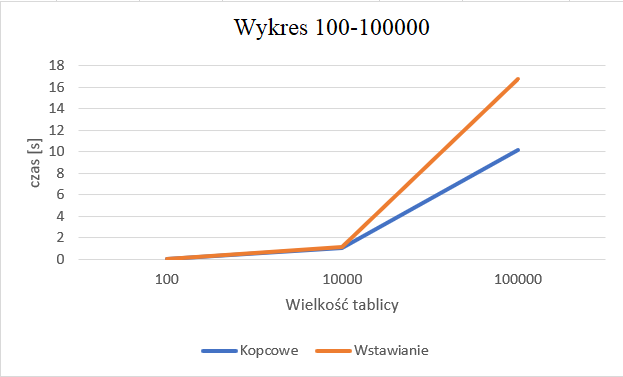


Tabela 1 Zależność czasu od wielkości tablicy



Wykres 1 Zależność czasu od wielkości tablicy

# **Pseudokod**

## 4.1 Lista kroków dla algorytmu sortowania przez kopcowanie

### Budowa kopca:

K01: Dla i = 2, ..., n:

wykonuj kroki K02...K05

K02: j ← i; k ← j div 2

K03: x ← tab[i]

K04: Dopóki (k > 0) ∧ (tab[k] < x):

wykonuj:

tab[j] ← tab[k]

j ← k

k ← j div 2

K05: tab[j] ← x

K06: Zakończ

* + 1. Rozbiór kopca

K01: Dla i = n, n - 1, ..., 2:

wykonuj kroki K02...K08

K02: tab[1] ↔ tab[i]

K03: j ← 1; k ← 2

K04: Dopóki (k < i):

wykonuj kroki K05...K08

K05: Jeżeli (k + 1 < i) ∧ (tab[k + 1] > tab[k]),

to m ← k + 1

inaczej m ← k

K06: Jeżeli tab[m] ≤ tab[j],

to wyjdź z pętli K04

i kontynuuj następny obieg K01

K07: tab[j] ↔ tab[m]

K08: j ← m; k ← j + j

K09: Zakończ

## 4.2 Lista kroków dla algorytmu sortowania przez wstawianie

K01: Dla j = n - 1, n - 2, ..., 1:

wykonuj kroki K02...K04

K02: x ← tab[j]; i ← j + 1

K03: Dopóki ( i ≤ n ) ∧ ( x > tab[i] ):

wykonuj tab[i - 1] ← tab[i]; i ← i + 1

K04: tab[i - 1] ← x

K05: Zakończ

# **Wnioski**

Projekt został zrealizowany, działanie algorytmu jest poprawne. W programie główny algorytm został zaimplementowany w osobnej funkcji, która jest wywoływana w późniejszych etapach działania programu. Projekt posiada możliwość odczytu z pliku tekstowego jak i późniejsze zapisanie wyników do pliku tekstowego. W programie zostały umieszczone stosowane komentarze, które pomagają w zrozumieniu kodu również w sprawozdaniu zostały umieszczone elementy które ułatwiają nam zrozumienie działania algorytmu szukającego jest to pseudokod i schemat blokowy.