Laboratoria Algorytmów i Struktur Danych

Projekt 2 - sprawozdanie

# Drzewa przeszukiwań binarnych BST i drzewa samobalansujące


Prowadzący: **Dominik Piotr Witczak**

Przygotowali: **Jan Czyżewski 160377, Albert Łapa 160381**

Grupa: **LAB 13**

Kierunek: **Informatyka, zajęcia pt. 8:00-9:30**


POLITECHNIKA POZNAŃSKA

# 1 | Menu

Program wyposażono w przejrzyste menu **help**. Widać w nim każdą z zaimplementowanych funkcji operacji na drzewach.

```
PS C:\Users\alber\Desktop\AiSD> & C:/Users/alber/AppData/Local/Programs/Python/Python312/python.exe c:/Users/alber/Desktop/AiSD/main.py -avl
command> help
--- Help ---
Commands:
Help - display this message
Exit - exit the program
Print - print the trees
Insert - insert a node into the trees
Delete - delete all nodes from the trees
Rebalance - rebalance the AVL tree
Remove - remove a node from the trees
FindMinMax - find the minimum and maximum values in the trees
SortAndMedian - sort the trees and find the median
Tickz - save the tree to a txt file
------------
command>
```

Rysunek 1: Menu programu

# 2 | Operacje na drzewach

Poniższe zrzuty pokazują działanie dostępnych funkcji programu.

```
command> insert
nodes> 7
insert> 2 5 10 12 13 6 9
command> print
AVL  tree:
In-order: 2 5 6 9 10 12 13
Post-order: 2 9 6 5 13 12 10
Pre-order: 10 5 2 6 9 12 13
command> findminmax
AVL tree: Min = 2, Max = 13
command> sortandmedian
AVL tree sorted:
2 5 6 9 10 12 13
Median: 9
command>
```

Rysunek 2: Funkcje programu

(a) Wprowadzanie elementów do drzewa.

(b) Wyświetlanie elementów drzewa, posegregowanych każdą z metod.

(c) Wyszukiwanie elementów największego i najmniejszego w drzewie.

(d) Wyświetlanie posortowanych elementów drzewa od najmniejszego do największego.

```
command> remove
remove> 2 5 10
command> print
AVL  tree:
In-order: 6 9 12 13
Post-order: 9 6 13 12
Pre-order: 12 6 9 13
command> delete
All nodes have been deleted from the AVL tree.
command> print
AVL  tree:
In-order:
Post-order:
Pre-order:
```

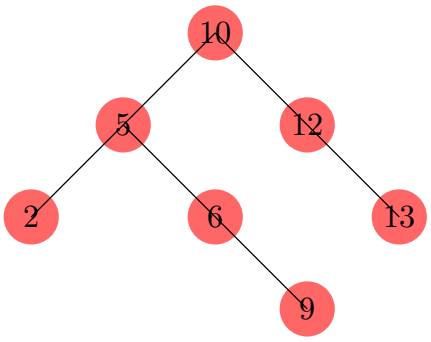
Rysunek 3: Funkcje programu cd

(e) Usuwanie elementów o podanych przez użytkownika wartościach.

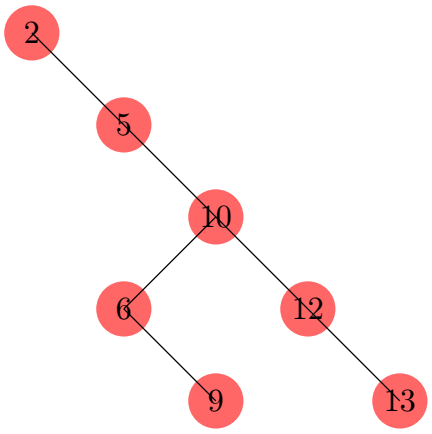
(f) Usuwanie całego drzewa.

```
PS C:\Users\alber\Desktop\AiSD> & C:/Users/alber/AppData/Local/Programs/Python/Python312/python.exe c:/Users/alber/Desktop/AiSD/main.py -avl
command> insert
nodes> 7
insert> 2 5 10 12 13 6 9
command> tickz
AVL tree has been saved to a txt file.
command> exit
PS C:\Users\alber\Desktop\AiSD> & C:/Users/alber/AppData/Local/Programs/Python/Python312/python.exe c:/Users/alber/Desktop/AiSD/main.py -bst
command> insert
nodes> 7
insert> 2 5 10 12 13 6 9
command> tickz
BST tree has been saved to a txt file.
command>
```

Rysunek 4: Eksportowanie drzew w formie kodu LaTeX do pliku tekstowego




(a) Wyeksportowane drzewo AVL



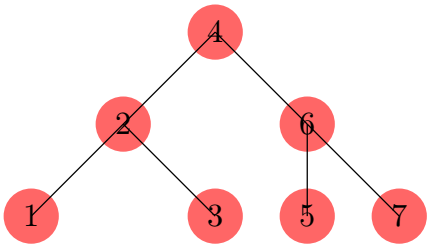

(b) Wyeksportowane drzewo BST

```
PS C:\Users\alber\Desktop\AiSD> & C:/Users/alber/AppData/Local/Programs/Python/Python312/python.exe c:/Users/alber/Desktop/AiSD/main.py -avl
command> insert
nodes> 7
insert> 1 2 3 6 5 4 7
command> print
AVL  tree:
In-order: 1 2 3 4 5 6 7
Post-order: 1 2 4 7 6 5 3
Pre-order: 3 2 1 5 4 6 7
command> rebalance
AVL tree has been balanced.
command> print
AVL  tree:
In-order: 1 2 3 4 5 6 7
Post-order: 1 3 2 5 7 6 4
Pre-order: 4 2 1 3 6 5 7
command> tickz
AVL tree has been saved to a txt file.
command>
```

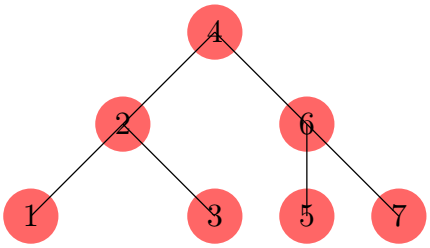
Rysunek 6: Balansowanie drzewa AVL

```
PS C:\Users\alber\Desktop\AiSD> & C:/Users/alber/AppData/Local/Programs/Python/Python312/python.exe c:/Users/alber/Desktop/AiSD/main.py -bst
command> insert
nodes> 7
insert> 1 2 3 6 5 4 7
command> print
BST  tree:
In-order: 1 2 3 4 5 6 7
Post-order: 4 5 7 6 3 2 1
Pre-order: 1 2 3 6 5 4 7
command> rebalance
BST tree has been balanced.
command> print
BST  tree:
In-order: 1 2 3 4 5 6 7
Post-order: 1 3 2 5 7 6 4
Pre-order: 4 2 1 3 6 5 7
command> tickz
BST tree has been saved to a txt file.
command>
```

Rysunek 7: Balansowanie drzewa BST




(a) Zbalansowane drzewo AVL

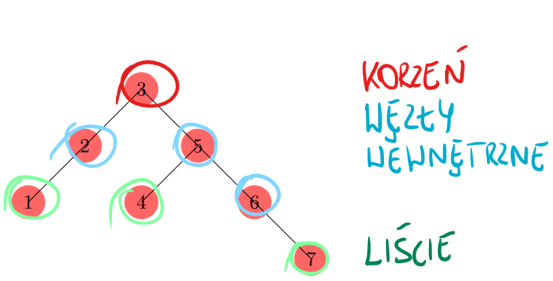



(b) Zbalansowane drzewo BST

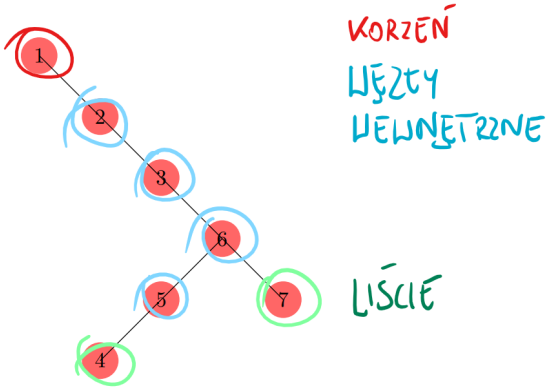
# 3 | Wizualizacje

```
nodes> 7
insert> 1 2 3 6 5 4 7
```

Rysunek 9: Dane, dla których wykonano wizualizacje




(a) Drzewo AVL - wizualizacja

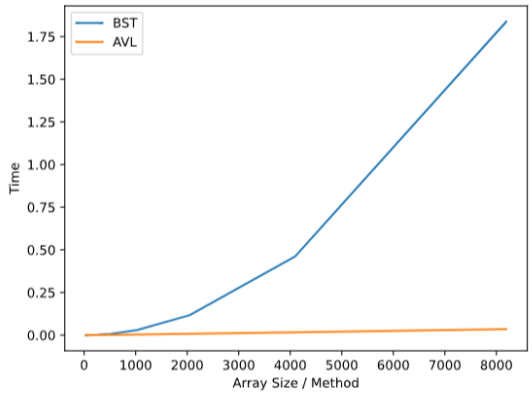



(b) Drzewo BST - wizualizacja

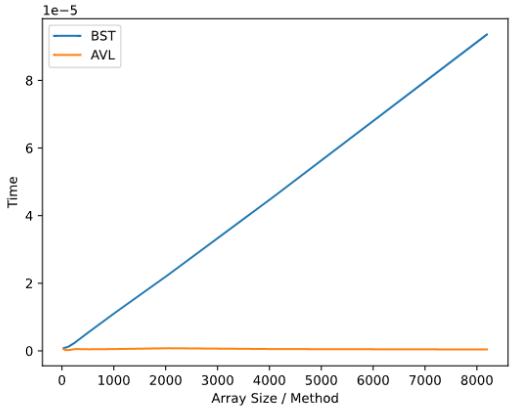
# 4 | Wykresy

Efektywność algorytmów sprawdzono przy pomocy dostarczonych plików typu benchmark. Komputer testujący wyposażony był w jednostkę obliczeniową Apple M1 i 16 GB pamięci operacyjnej. Pracował pod kontrolą systemu operacyjnego MacOS Sonoma 14.4.1. Testy wykonano na algorytmach napisanych w języku Python w wersji 3.12.2.

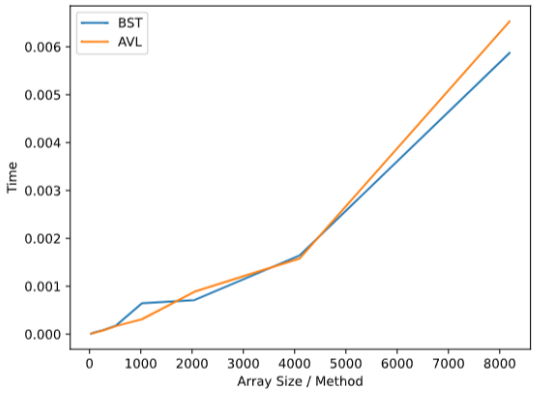
## 4.1 | Dla tablicy wartości zdegenerowanych



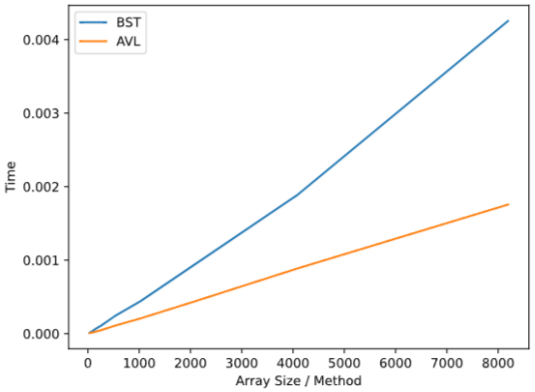

Tworzenie struktury




Wyszukiwanie min/max

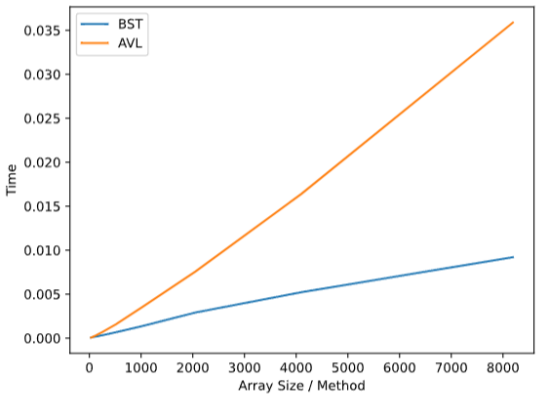



Wypisanie in-order

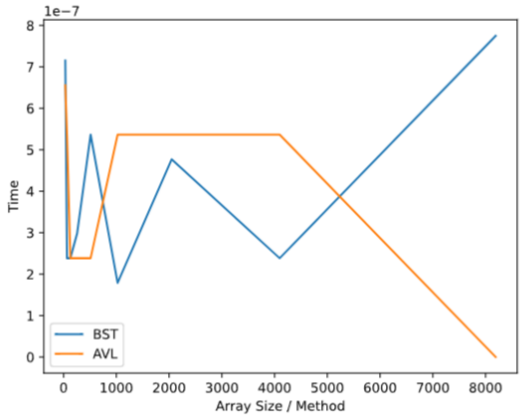



Równoważenie drzewa BST

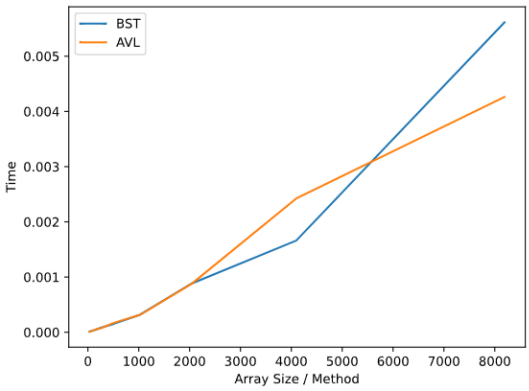
## 4.2 | Dla tablicy wartości losowych



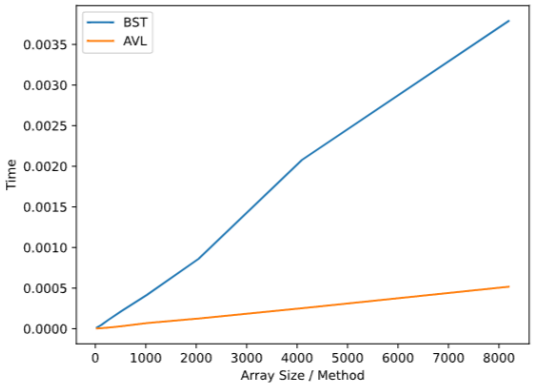

Tworzenie struktury




Wyszukiwanie min/max




Wypisanie in-order




Równoważenie drzewa BST

# 5 | Podsumowanie

Powyższe wykresy pokazują różnicę w zachownianiu omawianych algorytmów podczas używania różnych typów danych. Obrazują również, kiedy lepiej zastosować drzewo AVL, a kiedy BST.