**Львівський Національний Університет імені ІВАНА ФРАНКА**

Факультет прикладної математики та інформатики

**Алгоритми і Структури Даних**

**ЛАБОРАТОРНА РОБОТА №11**

**AVL Дерево.**

Виконав:

**Заяц Ростислав Васильович**

Кафедра Програмування. Група ПМО-11с

Оцінка:

Перевірив:

2025

**Теорія.**

**AVL-дерево** — це самобалансуюче бінарне дерево пошуку (Binary Search Tree, BST), яке гарантує, що висота дерева завжди залишається майже рівною. Воно отримало свою назву на честь його творців — **Адельсона-Вельського** і **Ландіса**, які запропонували цю структуру в 1962 році.

**Реалізація.**

Для реалізації програми для роботи з AVL-дерево оголосимо структуру яка буде моделювати дерево

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Дерево складається з корня, який зберігає значення, та з лівого та правого піддерева. Для комфортної роботи створимо дві функції.

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Одна дає нам доступ до висоти, інша показує баланс дерева. Реалізуємо їх за допомогою простого тернарного оператора.

Для того щоб правильно балансувати дерево нам потрібні rotate функції, які будуть обертати дерево в одну з сторін.

Зображення, що містить текст, знімок екрана, програмне забезпечення, Шрифт

Автоматично згенерований опис

Обертання змінює структуру дерева, переміщуючи вузли з однієї сторони на іншу, але зберігаючи порядок вузлів, який важливий для властивостей бінарного дерева пошуку. Наприклад для правого обертання воно працює так:

Зображення, що містить схема, ряд, дизайн

Автоматично згенерований опис

Нехай ключ 20 = X, а ключ 30 = Y. Тоді X стає новим коренем, зберігаючи зв'язок зі своїм лівим сином, Y стає правим сином нового кореня, а праве піддерево X стає лівим піддеревом вузла Y.

Обовязково після обертання обновити висоту. Висота вираховується за формулою 1(поточна висота) + максимальна висота лівого і правого піддерева (Умновно максимальна кількість синів лівого чи правого піддерева + сам вулоз = висота)

**Вставка**

**Зображення, що містить текст, знімок екрана, програмне забезпечення

Автоматично згенерований опис**

Головна операція у дереві, ми перевіряємо чи існує корінь у дерева, якщо ні створюємо його. Якщо значення менші за корінь, то за принципом бінарного дерева пошуку воно прямує у ліве піддерево, тому перевіряємо чи існує ліве піддерево, якщо ні то створюємо, якщо так, то то викликаємо рекурсивно метод вставки для лівого піддерева. Ідентично з правим. Наспутним кроком ми обновляємо нашу висоту, потім перевіряємо збалансованість дерева. Допустимі межі -1 до 1. Якщо у нас баланс кореня більше за 1 і значення вставляється в самий лівий листок, це означає що ліве піддерево у нас важче тому ми виконуємо праву ротацію. Ідентично з правим піддеревом. Якщо ж баланс більше 1 але елемент вставляється в праве піддерева лівого дочірного вузла тоді нам спочатку потрібно зробити ліву ротація для лівого дочірного вузла, а після цього праву. Ідентично з правим піддеревом.

**Main.**

Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Автоматично згенерований опис

**G-Test**

**Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис**