|  |  |  |  |
| --- | --- | --- | --- |
| МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка  ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  **Кафедра програмних систем і технологій**  Дисципліна  **«Структури даних, аналіз і**  **алгоритми комп’ютерної**  **обробки інформації»**  **Лабораторна робота № 2**  **на тему:**  **Бінарні дерева** | | | |
| **Виконав:** | Мичковський  Богдан  Ігорович | **Перевірила**: | Бичков Олексій Сергійович |
| Група | ІПЗ-12(1) | Дата перевірки |  |
| Форма навчання | денна | Оцінка |  |
| Спеціальність | 121 |
| 2022 | | | |

**1) Умова задачі:**

Використовуючи структуру даних: бінарне дерево пошуку, червоно-чорне дерево та АВЛ дерево реалізувати такі алгоритми:

1. Вставка вузла в дерево

2. Видалення вузла із дерева

3. Перефарбування двох однакових вузлів братів у протилежний колір

4. Лівий та правий повороти

5. Splay алгоритм

**2) Аналіз задачі:**

Перед початком виконання лабораторної роботи, було проведено повторення лекційного матеріалу та матеріалу з практик. Проведено попередній аналіз структур даних та алгоритмів виконання завдань.

Бінарне дерево – це структура даних у вигляді дерева, в якій кожен елемент не може мати більше двох нащадків, та містить посилання на батька у випадку видалення елемента з дерева. Доповненнями бінарного дерева є АВЛ дерево, червоно-чорне дерево та Splay дерево.

Червоно-чорне дерево – це структура даних у вигляді дерева, яка є само балансуючою та балансується за допомогою властивостей кольорів для регулювання відносної збалансованості при додаванні та видаленні елементів.

АВЛ дерево – це структура даних у вигляді дерева, яка є само балансуючою та максимальна висота її під дерев не перевищує 1.

Splay дерево – це структура даних у вигляді дерева, яка є розширенням бінарного дерева та є само балансуючою. При додаванні елемента або видаленні елемента на місце кореня стає елемент, який додається або елемент який є предком видаленого елемента.

**3) Структура основних вхідних та вихідних даних:**

Вхідними даними є будь-які числа, або числа підготовані завчасно.

Вихідними даними є видозмінені дерева після додавання або видалення елемента з дерева та операції розширених дерев застосованих до них.

**4) Алгоритм розв’язання задачі:**

Бінарне дерево:

Створення класу Node та створення класу дерево. Клас Node складається з цілочисельного типу даних який мітить значення яке ми зберігаємо та посилання на нащадків та предка кожного вузла (корінь батька немає).

При додаванні елемента до дерева використаємо властивість бінарного дерева, тобто якщо елемент менше поточного ми переходим до лівого вузла, якщо навпаки – переходимо до правого вузла, поки не дійдемо до не існуючого вузла, в який і збережем нове значення.

При пошуку максимального та мінімальних значень ми йдемо до кінця дерево (до крайнього лівого нащадка та крайнього нащадка відповідно) та виводимо цей елемент на консоль.

При пошуку елемента в дереві, використовуючи ті самі властивості що й при додаванні, при знаходженні елемента виводимо його на консоль, якщо ні виводимо що елемента не існує.

При видаленні елемента використаємо метод пошуку елемента для видалення, потім знайдемо елемент для заміни цього елемента для кожного під дерева по-різному, та збережемо посилання на них. Після цього можемо провести заміну елементів для видалення, незабуваючи передати елементу яким заміняєм нових нащадків та предка (за необхідності).

Під час захисту лабораторної роботи було додано перевірку на те, чи елемент є коренем та методо його подальшого видалення.

АВЛ дерево:

Створимо клас Node та клас АВЛ дерева. Значення всіх об’єктів параметрів та змінних як і в бінарного дерева та добавим додатково параметр висоти вузла.

Операції пошуку елемента, максимального та мінімального значення залишаємо незмінними.

При додаванні елемента повторюємо всі операції як і в звичайному бінарному дереві, проте після кожного етапу переходу до наступного елемента добавляємо перевірку на вірність правила висоти. Виконуючи певні операції обертів корегуємо розташування елемента відносно дерева. За необхідності міняємо корінь дерева на відповідний елемент.

При видаленні елемента проводиться перевірка на висоту під дерева та відповідне регулювання напрямку переформування дерева загалом.

Червоно-чорне дерево:

Створимо клас Node та клас червоно-чорного дерева. Значення всіх змінних співпадає зі значеннями параметрів та змінних бінарного дерева та містить змінну, яка визначає колір дерева (true - червоне, false - чорне).

Операції пошуку елемента, максимального та мінімального значень співпадають. При додаванні елемента проводиться перевірка на розташування даного елемента в дереві та за потреби його відсортування. Дальше ми добавляємо елемент та перевіряємо чи всі елементи самого дерева не містять двох червоних вузлів підряд.

Splay дерево:

Створимо клас Node та клас Splay дерева. Значення всіх змінних співпадає зі значеннями параметрів та змінних бінарного дерева та не потребує ніяких доповнень.

Операції пошуку елемента, максимального та мінімального значень співпадають, проте при операції пошуку елемента за значенням, ми замінюємо цим елементом корінь дерева та перетворюємо ціле дерево відповідно до властивостей Splay дерева.

При видаленні об’єкта проводимо перевірку на наявність елемента для видалення, потім, якщо елемент містить нащадків, шукаємонащадка для заміни. Після заміни елемента перевіримо чи всі елементи мають правильні кольори.

**5) Текст програми – додано на GitHub.**

**6) Набір тестів:**

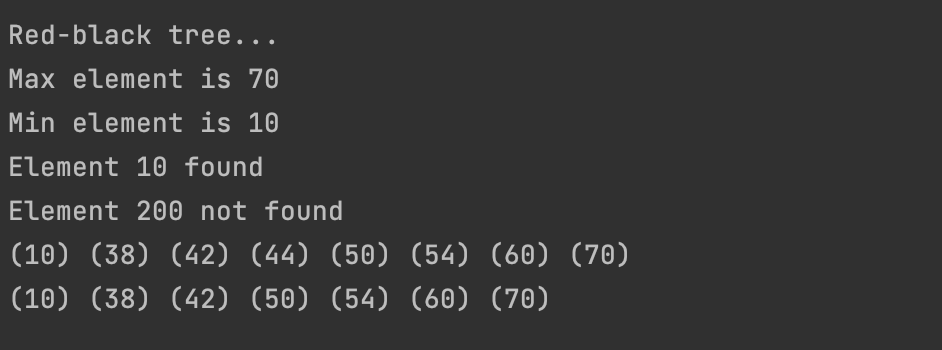
В різних тестах проведем різні можливі варіанти при яких програма може викинути помилку. Видалення елемента в корені, заміна елемента в корені, пошук існуючих та неіснуючих елементів, перевірку на пустоту дерева, та перевірка результатів виконання програми. Нище наведені приклади вище описаних тестів.

Text

Description automatically generated

Text

Description automatically generated



Text

Description automatically generated

**7) Результат тестування програми та аналіз помилок:**

В результаті виконання вище описаних тестів програми не видала ніяких помилок, та вивела всю потрібну інформацію, та успішно видалила всі елементи описані в коді програми.

Під час виконання лабораторної роботи особливо багато складностей мені доставило написання методів видалення елемента. Незважаючи на розуміння потрібного виконання програми, можливі варіанти вирішення частково могли бути незадовільними для виконання програмою видалення на великих деревах, проте всі проблеми були вирішені або не виникла помилка яка могла перервати виконання програмою необхідних дій.

Через деякі труднощі з додаванням та видаленням елемента під час виконання операцій червоно-чорного дерева, були проблеми з повною відсутністю сортування елементів та вузлів, перефарбовування елементів дерев під час додавання або видалення та зникнення деяких окремих нащадків після операцій.

Під час здачі лабораторної роботи, було виявлено помилку при видаленні кореневого елемента. Проблему було вирішено додаванням окремої перевірки для того чи є елемент для видалення коренем та проведення відповідних операцій для заміни кореневого елемента.

**Висновок:**

Під час виконання лабораторної роботи мною було сформовано навички використання бінарних дерев, їх модифікацій, доповнень та розширень з метою реалізації різних алгоритмів пошуку. Використання алгоритмів звичайного бінарного дерева та Splay дерево були не складними, так як вони мають дуже подібну структуру, та результат виконання програми є очевидним. Червоно-чорне дерево викликало в мене деякі проблеми в реалізації алгоритмів, тому червоно-чорне дерево буде готове найближчим часом.