Київський національний університет імені Тараса Шевченка

Факультет комп’ютерних наук та кібернетики

Звіт

лабораторної роботи №6

«B+ дерево»

з предмету «Алгоритми та складність»

Над роботою працював:

студент 2 курсу

групи К-29

Маханько Ростислав

2019

**Завдання**. Реалізувати B+ дередо.

**Вхідні дані**: У першому рядку файлу з вхідними даними міститься число команд. Далі для кожної команди її назва, кількість гравців та їх імена, кожне значення з нового рядка. З клавіатури зчитаємо для кожної команди її ключ, який представлений цілим числом. Кожна команда повинна мати унікальний ключ.

**Вихідні дані(взаємодія з інтерфейсом)**: Існує чотири типи запитів:

1. Додати нову команду. Зчитуємо назву команди та інформацію про її гравців, а також ключ цієї команди. Якщо команди з таким ключем ще не було, вставляємо її в дерево.
2. Знайти команду за ключем. Якщо така команда є, то виведемо її назву, інакше – рядок “NO”.
3. Видалити команду за ключем. Якщо команди з таким ключем немає відповідне повідомлення буде відображене на екрані.
4. Виведення B+ дерева на екран.

**Алгоритм розв’язання**: B+ дерево — тип дерева, яке подає відсортовані дані в вигляді, що дозволяє швидке додавання, отримання і видалення записів, кожен з яких ототожнений ключем. Це динамічний, багаторівневий індекс, з верхньою та нижньою межами на кількість ключів в кожному сегменті індексу(блоці або вершині). В B+ дереві, на відміну від B дерева, всі записи зберігаються на рівні листових вузлів дерева; у внутрішніх вузлах зберігаються лише ключі.

Головне призначення B+-дерева — в зберіганні даних для швидкого отримання в блоково-орієнтованих сховищах — особливо, файлових системах. Це здебільшого через те, що на відміну від бінарного дерева пошуку, B+ дерева мають дуже значне розгалуження (зазвичай близько 100 й більше), що зменшує кількість операцій введення-виведення потрібних для знаходження елемента в дереві.

У B-дереві у всіх вершинах зберігаються ключі разом з додатковою інформацією. У B + -дерева вся інформація зберігається в листах, а у внутрішніх вузлах зберігаються тільки копії ключів. Таким чином вдається отримати максимально можливу ступінь розгалуження у внутрішніх вузлах. Крім того, лист може включати в себе покажчик на наступний лист для прискорення послідовного доступу, що вирішує одну з головних проблем B-дерев.

У даній реалізації у вузлі зберігається: кількість ключів вузла; самі ключі; вказівники на дітей або, якщо вузол є листом, на значення; вказівник на батька; інформація, чи є вузол листом. Максимальна степінь вузла дерева B=31.

Операції:

* findLeaf

Допоміжна функція, яка повертає лист, в якому повинен знаходиться переданий їй ключ. Визначаємо інтервал і переходимо до відповідного сина. Повторюємо поки не дійшли до листа.

* insert

Шукаємо лист, в якому можна додати ключ і додаємо його до списку ключів. Якщо вузол не заповнений, то додавання завершено. Інакше розбиваємо вузол на два вузла. Будемо вважати, що в дереві не може перебувати 2 однакових ключа, тому insert буде повертати чи був доданий ключ.

* split

Розбиття на два вузла відбувається наступним чином: в перший додаємо перші (B–1)/2 ключів, в другий ті що залишились. Якщо вузол - лист, то ключ, що залишився також додається в праве піддерево, а його копія відправляється в батьківський вузол, де стає точкою розподілу для двох нових піддерев.

Якщо і батьківський вузол заповнений - робимо аналогічно, але не копіюємо, а просто переміщаємо ключ, що залишився в батьківський вузол, так як це просто копія. Повторюємо поки не зустрінемо незаповнений вузол або не дійдемо до кореня. В останньому випадку корінь розбивається на два вузла і висота дерева збільшується.

Оскільки до батька завжди відправляється мінімальний ключ з другої половини, то кожен ключ, який зберігається у внутрішній вершині - це мінімум правого піддерева для цього ключа.

* erase

Оскільки всі ключі знаходяться в листах, для видалення в першу чергу необхідно знайти лист, в якому він знаходиться. Якщо вузол містить не менше (B–1)/2 ключів, то видалення завершено. Інакше необхідно виконати спробу перерозподілу елементів, тобто додати в вузол елемент з лівого або правого брата (не забувши оновити інформацію в батьку). Якщо це неможливо, необхідно виконати злиття з братом і видалити ключ, який вказує на видалений вузол. Об'єднання може поширюватися на корінь, тоді відбувається зменшення висоти дерева. Так як ми вважаємо, що в дереві не може перебувати 2 однакових ключів, то erase буде повертати чи був видалений ключ.

Нехай t=(B–1)/2 – мінімальна степінь дерева, тоді всі вищенаведені операції мають складність O(t\*logtn).

**Використана література**:

[https://uk.wikipedia.org/wiki/B%2B\_дерево](https://uk.wikipedia.org/wiki/B%2B_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE)

<https://en.wikipedia.org/wiki/B%2B_tree>

<https://neerc.ifmo.ru/wiki/index.php?title=B%2B-%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE>

[https://uk.wikipedia.org/wiki/Б-дерево](https://uk.wikipedia.org/wiki/%D0%91-%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE)