Київський національний університет імені Тараса Шевченка

Факультет комп’ютерних наук та кібернетики

Лабораторна робота №8

Виконав: Олексишин Олексій

Група К-28

Кафедра інтелектуальних програмних систем

**2020**

**Предметна область**

Вариант 3

Предметная область Отдел кадров Объекты Подразделения, Сотрудники Примечание Имеется множество подразделений предприятия. В каждом подразделении работает множество сотрудников.

Для того щоб дану структуру даних можна було помістити в дерево я перевантажив operator< (та інші оператори порівняння), для виведення в консоль operator<<.

**Завдання**

Реалізуйте структуру даних фіббоначієва піраміда.

**Алгоритм і теорія**

Фібоначчієва піраміда – набір дерев, для кожного з яких виконується властивості піраміди (корені дерев зберігаються у двозв’язному списку). Перевагою фібоначчієвої піраміди є те що всі операції, які не потребують видалення елементів працюють за амортизованою часовою складністю O(1), а інші за O(log(n)) як і в звичайних пірамідах. Операція merge злиття двох пірамід теж виконується за O(1).

У кожного вузла піраміди крім значення є також атрибути left, right, parent, child, degree (кількість дочірніх вузлів), mark (показує чи втрачав вузол свої дочірні вузли після того як став сином якогось іншого вузла). Також піраміда зберігає вказівник на мінімальний елемент. Головна процедура, що допомагає зберігати властивості піраміди – Consolidate.

Merge

Оскільки всі корені дерев зберігаються у двозв’язному списку, то нам треба просто з’єднати ці два списки, а потім вибрати вказівник на мінімум як один із вказівників на мінімум у попередніх пірамід.

Insert

Додаємо просто до списку коренів новий корінь дерева, в який поміщаємо нове значення. За потреби змінюємо вказівник на мінімальне значення в піраміді.

GetMin

Просто повертаємо мінімальне значення, за вказівником на мінімальне значеня.

ExtractMin

Спочатку візьмемо за вказівником на мінімальний елемент сам мінімальни елемент, видалимо його з піраміди, всі його дочірні елементи додамо до списку дерев як нові дерева, змінимо вказівник на мінімум на вказівник на правий елемент від видаленого у списку, або NULL, якщо такого немає. Далі виконуємо процедуру Consolidate.

Consolidate

В цій процедурі ми створюємо масив A, в якому за індексом і буде зберігатися вершина, яка має рівно і синів. Проходимо по порядку по всьому дереву, якщо для якоїсь вершини ми знаходимо вершину, що має таку ж кількість синів, то робимо одну вершину (ту в якої більший ключ) сином іншої, продовжуємо поки в кількість синів у даної вершини не стане відмінною від всіх інших.

DecreaseKey

Знаходимо потрібний вузол і зменшуємо його ключ. Якщо ключ не менший ніж у батька, або ж вузол не має батька, то нічого робити не треба. Інакше видаляємо вузол з списку синів у його батька і вставляємо новою вершиною у список вузлів, встановлюємо у нього mark = false. Якщо нове значення є меншим за мінімальне у піраміді, то треба ще й змінити вказівник на мінімум. Далі повторюємо рекурсивно дану операцію з його батьком: якщо у нього є батько і для нього mark=false, то встановлюємо mark=true, видаляємо цей елемент у його батька, вставляємо у список вершин, і так далі.

Delete

Для початку знаходимо елемент, зменшуємо його до дуже малого значення (мінус нескінченність), він стає мінімумом, а потім просто робимо ExtractMin

**Складність алгоритму**

* Merge: O(1)
* Insert: максимальна – O(log(n)), середня – O(1) (амортизована)
* Minimum: O(1)
* ExtractMin: амортизована складність O(log(n))
* DecreaseKey: амортизована складність O(1)
* Delete: амортизована складність O(log(n))

**Реалізація (мова С++)**

Програма реалізує клас FibonacciHeap, що містить в собі реалізацію інтерфейсу для роботи з пірамідою Фібоначчі (insert, delete, і тд) та всі інші вказані вище процедури.

**Основні модулі програми**

У файлі fibonacci.hpp знаходиться реалізація структури даних та потрібні для роботи з ним методи (delete\_key, decrease\_key, ectract\_min, insert, merge, min, consolidate) Файл struct.hpp містить реалізацію класів предметної області. В файлі main.cpp міститься інтерфейс користувача та приклад роботи з реалізованою структурою даних.

Код можна знайти в моєму репозиторії на github за посиланням <https://github.com/ooleksyshyn/semester4/tree/master/algorithms/lab8>

**Інтерфейс користувача та приклади**

Програма дозволяє взаємодіяти з класом через простий інтерфейс – вставка, видалення, пошук елемента за ключем та. В коді є наведений приклад використання програми.

**Літературні джерела**

* [https://ru.wikipedia.org/wiki/Ф](https://ru.wikipedia.org/wiki/Фибоначчиева_куча)ибоначчиева\_куча
* <https://github.com/ooleksyshyn/semester4/tree/master/algorithms/lab8>