Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Алгоритми та складність

Завдання №7

“Реалізація біноміальної піраміди”

Виконав студент 2-го курсу

Групи К-29

Пащенко Дмитро Вікторович

2020

**Завдання**:

Реалізувати біноміальну піраміду, застосувавши її до моделі, описаної у Вашому варіанті.

**Варіант 15:**

Предметна область: Залізниця

Об'єкти: Дороги, Станції

Примітка: Є безліч залізничних доріг. У відомстві кожної дороги знаходиться безліч станцій.

**Теорія:**

* Біноміальне дерево (англ. binomial tree) Bk — це рекурсивно означене упорядковане дерево.
* Біноміальне дерево Bk має наступні властивості:
  + має 2k вузлів;
  + має висоту k;
  + має рівно вузлів на глибині ;
  + має корінь ступеня k; ступінь всіх інших вершин менша ступеня кореня біноміального дерева. Крім того, якщо дочірні вузли пронумерувати зліва направо числами k - 1, k - 2, …, 0, то i-ий дочірній вузол кореня є коренем біноміального дерева Bi.
* Біноміальна купа (англ. binomial heap) — це множина біноміальних дерев, що задовольняє властивостям біноміальної купи:
  + Кожне біноміальне дерево у купі підпорядковується властивості неспадної купи (англ. min-heap property): ключ вузла не менший за ключ його батьківського вузла;
  + Для будь якого невід'ємного цілого k в купі існує не більше одного біноміального дерева,чий корінь має ступінь k.

**Мова програмування:** С++.

**Інтерфейс користувача**

Введення даних відбувається через консоль.

Вхідні дані: команди

* 0 – завершити програму
* 1 – вставити залізницю (за назвою)
* 2 – вилучити мінімальний ключ
* 3 – перевірити наявність ключа в піраміді
* 4 – видалити залізницю (за назвою)
* 5 – вивести піраміду

Результат: біноміальна піраміда залізниць.

**Модулі програми:**

* **void BinomialHeap<T>::Node::link(Node\* new\_child)**

Під'єднує зліва до даного вузла нового сина.

**Складність:** .

* **Node\* BinomialHeap<T>::findMin() const**

Повертає вузол з мінімальним ключем.

**Складність:** , де n – кількість вузлів піраміди.

* **void BinomialHeap<T>::merge(BinomialHeap<T> &that)**

Об'єднує корені даної піраміди з коренями вказаної, впорядковуючи їх в порядку зростання, причому вказана піраміда очищується, а утворена записується замість даної.

**Складність:** , де n – кількість вузлів піраміди.

* **void BinomialHeap<T>::unite(BinomialHeap<T> &to\_unite)**

Зливає вказану піраміду з даною, вказана піраміда очищується.

**Складність:** , де n – кількість вузлів піраміди.

* **Node\* BinomialHeap<T>::search(T const &key, Node\* current) const**

Шукає вершину за ключем.

**Складність:** , де n – кількість вузлів піраміди.

* **void BinomialHeap<T>::insert(T const &key)**

Вставляє ключ в піраміду.

**Складність:** , де n – кількість вузлів піраміди.

* **T BinomialHeap<T>::extractMin()**

Видаляє мінімальний ключ з піраміди й повертає його.

**Складність:** , де n – кількість вузлів піраміди.

* **void BinomialHeap<T>::decreaseKey(Node\* node, T const &new\_key)**

Змінює ключ вказаної вершини, якщо новий ключ менше.

**Складність:** , де n – кількість вузлів піраміди.

* **void BinomialHeap<T>::remove(T const &key)**

Видаляє ключ з піраміди.

**Складність:** , де n – кількість вузлів піраміди.

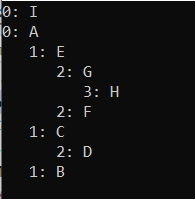
* **void BinomialHeap<T>::print() const**

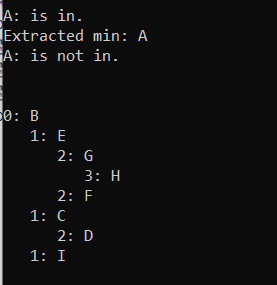
Друкує піраміду.

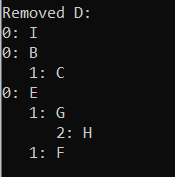
**Складність:** , де n – кількість вузлів піраміди.

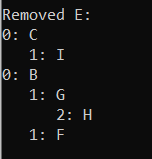
**Тестові приклади**

1. Створюю порожню піраміду. Задля зручності та незагромадженості називатиму залізниці великими буквами латинського алфавіту. Послідовно додав до піраміди ключі від «А» до «I». Бачимо, що властивості біноміальної піраміди виконуються. Можна легко переконатися в цьому, згадавши двійковий запис числа 9 (стільки вершин ми вставили) – 1001. Тобто піраміда має складатися з двох біноміальних дерев степенів В0 і В3, що ми й бачимо.



1. Далі перевіримо, чи є ключ «А» в піраміді – є. Вилучимо мінімальний ключ – отримали ключ «А», який і є мінімальним. Перевіримо наявність «А» в піраміді ще раз – тепер його вже немає. Виведемо піраміду. Бачимо, що всі властивості виконуються. Наразі кількість вершин – 8, тобто 1000. Наша піраміда – біноміальне дерево В3.
2. Видалимо ключ «D». Спочатку за лінійний час відбувається його знаходження в піраміді. Далі він заміняється на поточний мінімальний елемент – «В». Відбувається один обмін ключів – «В» і «С». Врешті решт, викликається процедура вилучення мінімального елемента – тобто, кореневої вершини «В». Отримуємо нову піраміду з 7 вершинами. 7 – це 111, тому наша піраміда це 3 біноміальних дерева: В0, В1 і В2.



1. Видаляю ключ «Е». За попередньо описаною процедурою, ключ «Е» стає ключем «В». Далі, за процедурою видалення, перше біноміальне дерево з коренем «В» вилучається з піраміди. Діти вершини «В» відокремлюються від батька й додаються в нову піраміду в порядку зростання степені. Нарешті відбувається злиття початкової й нової пірамід. Утворена піраміда має 6 ключів, що у двійковій системі 110, тому вона складається з біноміальних дерев степенів В1 і В2.

**Література**

* Кормен, Лейзерсон, Рівест, Штайн. Алгоритми: побудова і аналіз, 2-е видання. – 2005
* <https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%BD%D0%BE%D0%BC%D1%96%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE>
* <https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%BD%D0%BE%D0%BC%D1%96%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BA%D1%83%D0%BF%D0%B0>