Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Алгоритми та складність

Завдання №8

“Реалізація піраміди Фібоначі”

Виконав студент 2-го курсу

Групи К-29

Пупов Нікіта Андрійович

2020

**Завдання**:

Реалізація піраміди Фібоначі **.**

**Теорія:**

Піраміда Фібоначі - структура даних, що реалізує абстрактний тип даних «чергу з пріоритетом». Її властивості такі самі як і для звичайної піраміди. Її реалізація досить схожа на реалізацію біномінальної піраміди, за деяким вийнятком.

**Алгоритм**

При додаванні нової вершини(елемента) до піраміди, ми порівнюємо її з minNode – вказівником на вершину, що містить найменший ключ і оновлюємо його за необхідність(якщо нова вершина менша за minNode). Також цю вершину додаємо в масив roots(масив всіх коренів піддерев).

Таким чином, знаходження найменшої вершини досить швидке(ми повертаємо minNode).

На відміну від біномінальної кучі, тут ми не виконували операцію merge при додаванні нових елементів. Ми її виконуємо під час операції pop().

Таким чином, ми пришвидшуємо пошук найменшої вершини та додавання нової вершини за рахунок операції pop().

Операція merge порівнює елементи масива roots та обьеднує піддерева, корені яких мають однакову степінь.

При виконанні операції pop(), ми видаляємо зі структури minNode, виконуємо операцію merge та оновлюємо minNode(проходячи по масиву roots).

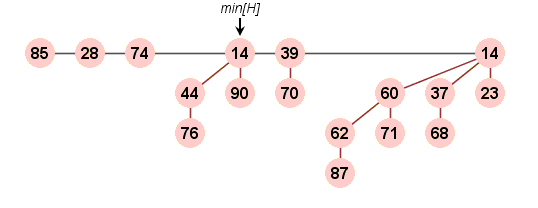


Рисунок 1 Можливий вигляд піраміди Фібоначі

**Мова програмування** С++

**Структура програми**

Програма складається з головного класу алгоритму FibonacciHeap, який працює з об’єктами типу File. Сортування відбувається за розміром файла.

**Модулі програми:**

* **void add(File file);**

Додає файл до піраміди.

Складність: О(1)

* **File top();**

Знаходить файл з найменшим розміром

Складність O(1)

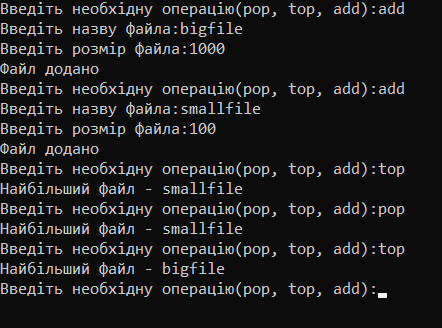
* **File pop();**

Знаходить файл з найменшим розміром та його видалення

Складність O(logn)

# Інтерфейс користувача

Користувачу доступні 3 операції: pop, top, add. Робота з інтерфейсом виглядає наступним чином:



Спочатку додаються 2 файли: bigfile, smallfile.

При виконанні операції top, ми отримуємо менший файл smallfile.

Виконуємо операцію pop, що знову повертає нам smallfile та видаляє його.

Тепер при виконанні операції top ми отримуємо bigfile, бо smallfile вже видалено.

# Висновок

За допомогою цієї структури, можна досить швидко виконувати операції знаходження мінімума серед багатьох елементів та додавати нові.

Таким чином, ми можемо реалізувати чергу з пріорітетом, беручи за основу дану структуру даних.

**Матеріали**

<https://www.youtube.com/watch?v=nZ0nFTvQez0>

<https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%B1%D0%BE%D0%BD%D0%B0%D1%87%D1%87%D0%B8%D0%B5%D0%B2%D0%B0_%D0%BA%D1%83%D1%87%D0%B0>