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

Пупов Нікіта Андрійович

2020

**Завдання**:

Реалізація біномінальної піраміди **.**

**Теорія:**

Біномінальна піраміда - структура даних, що реалізує абстрактний тип даних «чергу з пріоритетом», яка представляє собою набір біноміальних дерев з двома властивостями:

* ключ кожної вершини не менше ключа її батька
* кожним нащадком біномінальної піраміди є біномінальна піраміда.
* все біноміальні піддерева одного батька мають різний розмір
* степінь батька завжди більша степені синів

**Алгоритм**

Нехай є біномінальна піраміда. При додаванні елемента, ми створюємо вершину, в якій ключом є цей елемент. Ця вершина буде сином кореня основного дерева. Далі ми дивимося, якщо є такі сини кореня, в яких степені однакові – ми виконуємо операцію merge над ними.

Степенем біномінальнї піраміди назовемо кількість дочірніх вершин цієї піраміди.

Для виконання операції merge двох піддерев, ми обираємо піддерево з найменшим коренем і приєднуємо до нього(робимо сином) друге піддерево. Таким чином, у нас залишилося тільки одне піддерево зі степенем на один більше, ніж було. Повторюємо доки не залишаться піддерева з унікальними степенями.

Найменший елемент біномінальної піраміди знаходиться серед кореней усіх піддерев.

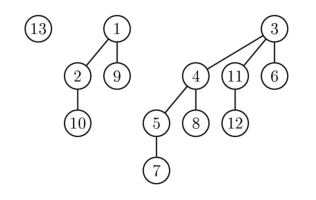


Рисунок 1 Тут зображено три біномінальні піраміди зі степенями 1, 2, 3 відповідно

**Мова програмування** С++

**Структура програми**

Програма складається з головного класу алгоритму BinominalHeap, 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(logn)

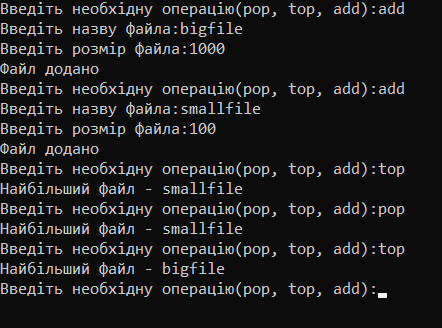
* **File pop();**

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=7UQd9SYUoNk>

<https://habr.com/ru/post/135232/>