Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Алгоритми та складність

Завдання №1

“Реалізація персистентної множини на основі бінарного дерева пошуку”

Виконав студент 2-го курсу

Групи К-29

Пупов Нікіта Андрійович

2020

**Завдання**:

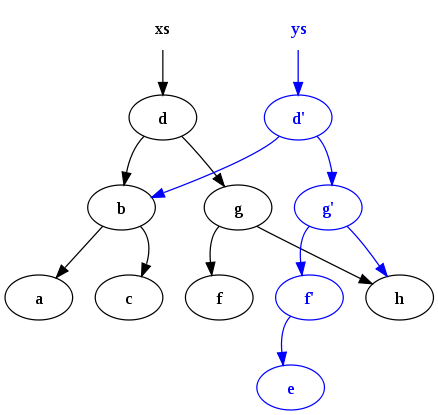
Реалізація персистентної множини на основі бінарного дерева пошуку **.**

**Теорія:**

Персистентні структури даних (англ. Persistent data structure) - це структури даних, які при внесенні в них якихось змін зберігають всі свої попередні стану і доступ до цих станів. Тобто, коли ми додаємо до множини новий елемент, сама множина не змінює свій стан, а створює нову множину зі зміненим станом.

**Алгоритм**

Нехай є бінарне дерево пошуку. Всі операції в ньому робляться за O (h), де h - висота дерева, а висота дерева O (logn), де n - кількість вершин. Нехай необхідно зробити якесь оновлення в цьому збалансованому дереві, наприклад, додати черговий елемент, але при цьому потрібно не втратити старе дерево. Візьмемо вузол, в який потрібно додати нового нащадка.

Замість того щоб додавати нового нащадка, скопіюємо цей вузол, до копії додамо нового нащадка, також скопіюємо всі вузли аж до кореня, з яких можна досягти перший скопійований вузол разом з усіма покажчиками. Всі вершини, з яких змінений вузол не можна досягти, ми не чіпаємо. Кількість нових вузлів завжди буде порядку логарифма. В результаті маємо доступ до обох версій дерева. До користувача повертаємо посилання но новий крінь.

**Мова програмування** С++

**Структура програми**

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

**Модулі програми:**

* **BinTree addFile(File file);**

Додає файл до дерева. Повертає нове дерево зі зміненним станом.

Складність: О(logn)

* **File getFile(string name);**

Знаходить файл по його імені.

Складність O(logn)

**Матеріали**

<https://neerc.ifmo.ru/wiki/index.php?title=%D0%9F%D0%B5%D1%80%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BD%D1%82%D0%BD%D1%8B%D0%B5_%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85>