Київський нацiональний унiверситет iменi Тараса Шевченка  
Факультет комп’ютерних наук та кiбернетики  
Кафедра iнтелектуальних iнформацiйних систем

Алгоритми та складнiсть  
Завдання №2  
"Дерево порядкової статистики"  
Виконав студент 2-го курсу  
Групи К-28  
Гуща Дмитро Сергійович

2020

**Завдання**

Реалізувати дерево порядкової статистики на основі червоно-чорного дерева.

**Теорія і Алгоритм**

Дерево порядкової статискики – це структура, що має час виконання основних операцій максимально наближений до логарифмічного, за який вона може знайти k-ий елемент у відсортованій за зростанням вибірці. Така складність отримується за допомогою балансування бінарного дерева пошуку. За задачу візьмемо реалізацію червоно-чорного дерева.

Нехай root – корінь піддерева, значення лівого піддерева меньші за батька, а значення правого – більше(за означенням). Якщо nLeft – кількість вершин в лівому піддереві root. Тому якщо nLeft + 1 = k, то значення в корні ж шуканою статистикою, інакше, якщо ключ шуканого елемента менше за ключ вузла root – ми переходимо в ліве піддерево вузла root (root = root->left) і там повторяємо попередню операцію. Інакше, якщо ключ шуканого елемента більший за ключ вузла root – ми переходимо в праве піддереву і шукаємо (k-nLeft-1)-шу порядкову статистику, оскільки вже є принайми (nLeft + 1) найменших елементів. В моєму випадку значеннями вузла є учбові группи(class Group), в яких в свою чергу є підгруппи студентів(class Student).

**Предметна область**

Варіант 4   
Предметна область:  Учбовий відділ   
Об’єкти:  Групи, Студенти  
Примітка:  Маємо множину учбових груп. Кожна група містить в собі множину студентів

**Складність алгоритму**

Червоно-чорні дерева — різновид збалансованих дерев, в яких за допомогою спеціальних трансформацій гарантується, що висота дерева *h* не буде перевищувати [O(*log n*)](https://uk.wikipedia.org/wiki/%D0%9D%D0%BE%D1%82%D0%B0%D1%86%D1%96%D1%8F_%D0%9B%D0%B0%D0%BD%D0%B4%D0%B0%D1%83). Зважаючи на те, що час виконання основних операцій на бінарних деревах (пошук, видалення, додавання елементу) є O(*h*), ці структури даних на практиці є набагато ефективнішими, аніж звичайні бінарні дерева пошуку.

**Мова программування**

С++

**Модулі програми**

Лабораторна складається з 4 .cpp файлів, та трьох класів реалізованих в кожному з цих файлів окрім main.cpp.

student.h

class Student{}; //Класс опису студента  
std::string getName(); // метод повертає ім’я студента  
void getStudent();//метод виводить ID та ім'я студента в консоль  
void setName(std::string name); //метод змінює ім'я студента

group.h

class Group {};  
Group() : title("NULL"); //конструктор пустої групи  
Group(std::string title); //конструктор з початковою назвою групи  
Group(std::string title, Student\* first\_student); //конструктор з початковою назвою групи та першим студентом  
std::string getGroupTitle(); //модуль повертає назву групи  
std::vector<Student\*> getGroupStudents(); //модуль повертає множину студентів  
void setGroupTitle(std::string title); //модуль змінює назву групи  
void setGroupStudents(std::vector<Student\*> students); //модуль змінює множину студентів  
void addStudent(Student\* student); //додати нового студента  
void printStudents(); //вивід у консоль усіх студентів групи

rbtree.h

struct Node {}; //структура вузла ЧЧ-дерева  
class RBTree {}; //класс ЧЧ-дерева  
RBTree() : root(nullptr); //конструктор пустого дерева  
Node\* getRoot(); //модуль повертає вказівник на корінь ЧЧ-дерева  
Node\* treeSearch(std::string titleToSearch, Node\* temp); //метод шукає групу за назвою (preorderWalk)  
void printStudents(Group\* groupToPrint); //виводить список студентів групи  
void insert(Group\* newGroup); //вставка нової групи до ЧЧ-дерева  
void repaintingInsert(Node\* nodeToRepaint); //перефарбування вузлів при вставці (за потреби)  
void leftRotate(Node\* nodeToRotate); //лівий поворот вузлів  
void rightRotate(Node\* nodeToRotate); //правий поворот вузлів  
void removeNode(Node\* parent, Node\* current, Group\* groupToRemove); // видалення вузла (другорядний метод)  
void remove(std::string titleToDelete, Node\* temp); // видалення вузла за групою (основний метод)  
void repaintingRemove(Node\* nodeToRepaint); // переарбування вузлів при видаленні (за потреби)  
void preorderWalk(Node\* temp); //прохід дерева префіксним методом (рекурсивно)  
void postorderWalk(Node\* temp); //прохід дерева постфіксним методом (рекурсивно)

main.cpp

void manu(); //модуль відповідає за вибір подальших дій  
void inputGroupAndStudents(Group& newGroup); //функція додавання нової групи та студентів  
void searchAGroup(RBTree educationalTree); //пошук групи у дереві  
void interactiveMode(RBTree& educationalTree); //інтерактивне використання программи  
int main(); //основна функція программи

**Інтерфейс користувача**

Користувач вводить дані в консоль. Дані виводяться в консоль.

**Тестові приклади**

|  |  |
| --- | --- |
| Input sequence | Output sequence |
| Input : | Output : |
| **K21, K23, K22** | K21, K22, K23 |
| Insertion:  K20, K24 | K24, K25 |

**Висновки**

Дерево порядкової статистики є один з підвидів бінарного дерева пошуку, яке додатково має операцію яка порівнює вузли за значенням і знаходить мінімальний, а також є додаткове поле розміру піддерева, яке можна використовувати для ідентифікації конкретного елемента. Дана структура може бути використана в буферах редакторів, та при знаходженні значень медіан.

**Література**

* <https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046jintroduction-to-algorithms-sma-5503-fall-2005/video-lectures/lecture-10-red-blacktrees-rotations-insertions-deletions/>
* <https://en.wikipedia.org/wiki/Order_statistic_tree>
* <https://www.geeksforgeeks.org/red-black-tree-set-2-insert/>
* <https://en.wikipedia.org/wiki/Red%E2%80%93black_tree>