Отчет о проделанной работе

Лабораторная работа №2

Дисциплина: Алгоритмы и структуры данных

Студент: Доронин С.Г.

Группа: ПРО-201в

# Задание на лабораторную работу

Сгенерировать 25 3-х значных неповторяющихся элементов.

Вывести их на экран.

Представить в виде идеально-сбалансированного дерева.

Вывести элементы дерева в обратном порядке.

# Входные и выходные данные

## Входные данные

array – сгенерированная последовательность:

251, 457, 797, 506, 904, 724, 870, 320, 912, 546, 735, 189, 941, 584, 729, 970, 125, 570, 542, 214, 974, 558, 123, 946, 270

## Выходные данные

**root** – узел корня дерева, от него можно добраться до любого элемента дерева.

Идеально-сбалансированное дерево:

.............................................251................................................

.....................457.............................................584........................

.........797.....................912.....................729.....................974............

...506.........870.........546.........189.........970.........542.........558.........946......

904...724...320...xxx...735...xxx...941...xxx...125...570...214...xxx...123...xxx...270...xxx...

**back\_array** – список состоящий из элементов дерева представленных в обратном порядке:

904, 724, 506, 320, 870, 797, 735, 546, 941, 189, 912, 457, 125, 570, 970, 214, 542, 729, 123, 558, 270, 946, 974, 584, 251

# Алгоритм построения дерева

@root **=** build\_ideal\_r**(**array**,** Node**.**new**(nil),** array**.**size**)** # Запускаем ф-ю построения дерева

# Рекурсивное построение Идеально-сбалансированного дерева

**def** **build\_ideal\_r(**list**,** node**,** count**)**

**unless** count **==** 0 # Если кол-во элементов для поддерева (count) != 0

**if** node**.**data**.**nil? # Если поле с данными у узла = nil, то это узел с дерева, с него начинается построение

node**.**data **=** list**.**first # Первый элемента из массива устанавливаем в качестве значения для поле данных

list**.**delete\_at**(**0**)** # Удаляем первый элемент из массива

**end**

node**.**left **=** build\_ideal\_r**(**list**,** Node**.**new**(nil),** count**/**2**)** # Левой ветке узла устанавливаем значение рез-та выпол-я этой функции с параметрами: (список эл-ов, новый узел, кол-ва элементов поддерва count/2)

node**.**right **=** build\_ideal\_r**(**list**,** Node**.**new**(nil),** count**-**count**/**2**-**1**)** # Тоже самое что и для левого поддерева, только кол-во элементов для поддерева: count-count/2-1

node # В качестве рез-та, ф-я возвращает текущий узел, который становится веткой другого узла

**end**

**end**