Слуцкий Никита. 053501. Лабораторная Работа №8. Реализация класса Set для long long

В основе моего Сета лежит структура Бинарное Дерево Поиска. В среднем все операции происходят за log(N) – N. Это происходит, потому что дерево не балансируется само => в худшем случае это всё равно работает за N. Но при хорошем раскладе сложность колеблется от логарифма до линии.

В классе присутствуют следующие приватные поля:

- root – указатель на «корень» дерева (первый) элемент
- _size – текущее количество элементов в Сете

В классе присутствуют следующие публичные и приватные методы:

- DeleteByPointer ( ) – удаляет по указателю узел в дереве *// приватный*
- RecursiveClear ( ) – для метода Clear ( ). Рекурсивно проходится по всем узлам и удаляет их *// приватный*
- RecursivePrint ( ) – для оператора вывода <<cout. Также рекурсивно обходит дерево *// приватный*
- GetMinimumValue ( ) – находит минимальный элемент в (под)дереве. Пока нигде не используется, а просто пара к аналогичному GetMaximumValue( ). Можно использовать для поиска минимума в Сете *// приватный*
- GetMaximumValue ( ) – находит максимальный элемент в (под)дереве. Используется для упрощения удаления узла с двумя дочерними узлами // приватный
- FindElement ( ) – находит указатель на узел по его значению. Используется для упрощения других методов *// приватный*
- Size ( ) – возвращает значение поля _size (без возможности изменять его, естественно) *// публичный*
- Has ( ) – проверяет, имеется ли переданное число в Сете. «Под капотом» сводится к FindElement ( ) *// публичный*
- Add ( ) – добавляет новый элемент (при отсутствии в текущем Сете) *// публичный*
- Delete ( ) – удаляет заданный (при наличии) элемент *// публичный*
- Clear ( ) – очищает весь Сет. Сводится к рекурсивному RecursiveClear ( ) *// публичный*

В классе переопределены операторы:

- Оператор потокового вывода <<cout, который выводит в виде: Set (n): { {i1} {i2} {i3} … {in} }. По аналогии с JS