ЗАДАНИЕ

Файл test.cpp - заготовка теста. Файл write.txt - последовательность строк для первоначальной вставки. Файл modify.txt - последовательность индексов для удаления и строк для вставки разделенные пробелом. Файл read.txt - последовательность индексов для чтения и строк для проверки разделенные пробелом. Алгоритм теста следующий - в заполненном контейнере данными из write.txt происходит последовательное обновление (удаление по индексу и вставка новой строки) записей из modify.txt и чтение (по заданному индексу с последующей проверкой) из read.txt. Контейнер должен обработать весь набор данных за время не превышающее 10 секунд (время актуально для процессора с тактовой частотой 2.6ГГц).

Здравствуйте, расскажу немного про мой алгоритм.

1. Я понимаю, что раз уж мы удаляем и получаем элементы по индексу, значит получение и должно быть со сложностью О(1). Также вставка и удаление должно выполняться достаточно быстро. В связи с этим я выбрал связку из 2-х контейнеров: vector и deque. Vector хранит внутри себя deque, за счет чего я могу получить объект за константное время.
2. Deque был выбран в связи с тем, что у него LegacyRandomAccecIterator и есть возможность удаления с конца и добавления в начало за О(1).
3. Так как в программе уже есть сами строки, я храню вместо копий строк, указатели на них внутри контейнера deque.

INSERT

Добавление элемента реализовано следующим образом:

1. проверяю, не заполнен ли последний контейнер deque. Если заполнен, выделяю новый.
2. Так как data\_ хранится в отсортированном виде, использую бинарный поиск, для нахождения позиции для вставки нового элемента
3. при нахождении вставляю, сложность вставки в deque О(VEC\_SIZE) в худшем случае

ERASE

Удаление элемента:

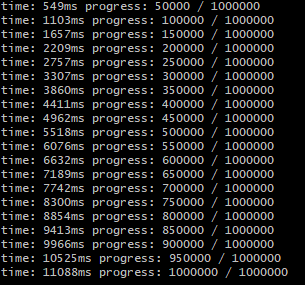
1. Так как data\_ хранится в отсортированном виде, нахождение нужного индекса выполняется за О(1), А удаление из deque за О(VEC\_SIZE). После этого привожу deque к нужному размеру, сложность О(M) , M - количество контейнеров deque в data\_.

GET

Получение элемента.

1. Так как data\_ хранится в отсортированном виде, нахождение нужного индекса выполняется за О(1).

На рисунке ниже приведены результаты тестирования



*рис. 1 Результаты тестирования*