Лабораторная работа №3

«Шаблоны и итераторы. Перегрузка операторов в классах»

Задание 1.

Реализовать статическую библиотеку Vector (аналог std::vector) не используя стандартные библиотеки. Использовать шаблоны. Реализовать простейший итератор и наследоваться от него в классе vector. Для работы класса итератора перегрузите операторы по аналогии с std::vector. В библиотеке vector необходимо реализовать следующие методы:

- assign; Удаляет вектор и копирует указанные элементы в пустой вектор.
- at; Возвращает ссылку на элемент в заданном положении в векторе.
- back; Возвращает ссылку на последний элемент вектора.
- begin; Возвращает итератор произвольного доступа, указывающий на первый элемент в векторе.
- capacity; Возвращает число элементов, которое вектор может содержать без выделения дополнительного пространства.
- cbegin; Возвращает *константный* итератор произвольного доступа, указывающий на первый элемент в векторе.
- clear; Очищает элементы вектора.
- data; Возвращает указатель на первый элемент в векторе.
- emplace; Вставляет элемент, созданный на месте, в указанное положение в векторе.
- emplace_back; Добавляет элемент, созданный на месте, в конец вектора.
- empty; Проверяет, пуст ли контейнер вектора.
- end; Возвращает итератор произвольного доступа, который указывает на конец вектора.
- erase; Удаляет элемент или диапазон элементов в векторе из заданных позиций.
- front; Возвращает ссылку на первый элемент в векторе.
- insert; Вставляет элемент или множество элементов в заданную позиции в вектор.
- max_size; Возвращает максимальную длину вектора.
- pop_back; Удаляет элемент в конце вектора.
- push_back; Добавляет элемент в конец вектора.
- rbegin; Возвращает итератор, указывающий на первый элемент в обратном векторе.
- rend; Возвращает итератор, который указывает на последний элемент в обратном векторе.
- reserve; Резервирует минимальную длину хранилища для объекта вектора.
- resize; Определяет новый размер вектора.
- size; Возвращает количество элементов в векторе.

- swap; Меняет местами элементы двух векторов.

Задание 2.

Реализовать struct pair. Предусмотреть вариант pair<pair<T, T>, pair<T, T>> a;

Разработайте оконное приложение используя собственную библиотеку vector. Необходимо создать объект класс vector, где каждый объект это pair< vector<int>, vector<pair<int,double>> >. Вывести две матрицы на экран, где первая матрица это первый аргумент pair (vector<int>), вторая – второй аргумент pair (vector <pair<int, double>>).

Задание 3.

Медианой последовательности с нечётным числом членов будем называть значение, которое встаёт в середину, если последовательность отсортировать. Т. е. половина значений последовательности не меньше медианного элемента и половина значений не больше медианы.

Для заданного вектора a построить вектор b из медиан подряд идущих троек элементов. Для неполных троек брать арифметическое среднее.

Пример.

Дано a = { 1, 5, 1, 4, 5, 6, 2, 1, 3, 4, 4, 4, 5, 7 }. Разбиваем на тройки: { 1, 5, 1 }, { 4, 5, 6 }, { 2, 1, 3 }, { 4, 4, 4 }, { 5, 7 }, последняя “тройка” неполная — два элемента. Получаем набор медиан (последнее значение — арифметическое среднее последних двух элементов): b = {1, 5, 2, 4, 6}.

Задание 4.

Продемонстрировать работу оставшихся методов вашего класса vector из задания 1 используя оконное приложение.