**array**

Моей задачей является реализации аналога std::array – массива статической длины (в отличие от vector)

Исходя из задания, я буду реализовывать свой класс MyArray. Класс шаблонный, как и стандартный массив.

Методы существующего массива взяты с cppreference.com. Логично предположить, что методы аналога должны быть аналогичны. (<https://en.cppreference.com/w/cpp/container/array>)

Класс MyArray

template <class T, size\_t N>

Методы, которые будут реализованы:

1. Constructor – конструктор. Все элементы инициализируются 0

MyArray(){};

1. Operator[] - обращение к элементу в указанной позиции, при помощи указателя или, что лучше, итератора +const iterator

Value\_type& operator[](int i) {};

1. Operator at() - обращение к элементу в указанной позиции, при помощи указателя или, что лучше, итератора, с проверкой границ +const iterator

iterator at(int i) {};

1. Empty – проверка, что size\_ == 0

bool empty(){};

1. Size – размер (количество элементов) (size\_)

int size() {};

1. Begin – начало итератора +const iterator

iterator begin() {};

1. End - конец итератора +const iterator

iterator end() {};

1. rBegin – наоборот +const iterator

reverseIterator rbegin() {};

1. rEnd - наоборот +const iterator

reverseIterator rend() {};

1. Вывод в стандартный поток (friend функция) (пользуемся итератором и выводим каждый элемент)

friend std::ostream& operator<< (std::ostream& out, const MyArray& arr);

1. Operator== - оператор равенства (по элементам) (проходим итератором, сравниваем элементы)

bool operator==(const MyArray& other){};

1. Operator!= - оператор неравенства (по элементам) (аналогично п.10)

bool operator!=(const MyArray& other){};

1. Operator> - оператор больше (по сумме элементов)

bool operator>(const MyArray& other){};

1. Operator< - оператор меньше (по сумме элементов)

bool operator<(const MyArray& other){};

1. Operator>= - оператор больше либо равно (по сумме элементов)

bool operator>=(const MyArray& other){};

1. Operator<= - оператор меньше либо равно (по сумме элементов)

bool operator<=(const MyArray& other){};

1. front() – указатель на первый элемент +const

T& front() {};

1. back() – указатель на последний элемент +const

T& back() {};

1. получение размера массива на этапе компиляции

constexpr int n = 10;

Поля класса:

1. data\_ - статический массив типа T
2. size\_ - Размер (для упрощения работы), тип size\_t

Codestyle: <https://style-guides.readthedocs.io/en/latest/cpp.html>

Обертка:

Будет реализована в виде цикла while с switch case – цифрами, идентифицирующими операцию:

1. Изменить элемент (ввести номер массива, ввести позицию, ввести новое значение)
2. Сравнить 2 массива == (ввести номер первого массива, ввести номер второго массива)
3. Сравнить 2 массива != (ввести номер первого массива, ввести номер второго массива)
4. Сравнить 2 массива >= (ввести номер первого массива, ввести номер второго массива)
5. Сравнить 2 массива <= (ввести номер первого массива, ввести номер второго массива)
6. Сравнить 2 массива >(ввести номер первого массива, ввести номер второго массива)
7. Сравнить 2 массива < (ввести номер первого массива, ввести номер второго массива)
8. Распечатать элемент массива (ввести номер массива, ввести позицию)
9. Проверить массив на пустоту (ввести номер массива)
10. Распечатать размер массива (ввести номер массива)
11. Распечатать первый элемент массива front() (ввести номер массива)
12. Распечатать первый элемент массива back() (ввести номер массива)
13. Распечатать массив (ввести номер массива)
14. Создать массив (ввести размер) массивы будут создаваться с именами arr+N, где N – порядковый номер, создать можно не более 5 массивов
15. Выйти из программы