**ADT Динамический массив ~ TODO абстрактный КЛАСС (общие методы)**

**Конструктор по умолчанию:**

**Вход: Нет.**

**Предусловия: Нет.**

**Процесс: Создает пустой динамический массив.**

**Выход: Нет.**

**Постусловия: Указатель на массив инициализирован значением nullptr, емкость установлена в 0, размер установлен в 0.**

**Конструктор с заданной начальной емкостью:**

**Вход: Целочисленное значение initialCapacity, указывающее начальную емкость массива.**

**Предусловия: initialCapacity должен быть неотрицательным целым числом.**

**Процесс: Создает пустой динамический массив с указанной начальной емкостью.**

**Выход: Нет.**

**Постусловия: Указатель на массив выделен с помощью оператора new, емкость установлена в initialCapacity, размер установлен в 0.**

**Деструктор:**

**Вход: Нет аргументов.**

**Предусловия: Нет.**

**Процесс: Освобождает память, выделенную под динамический массив.**

**Выход: Нет.**

**Постусловия: Указатель на массив освобожден, емкость и размер установлены в 0.**

**push\_back(const T& element):**

**Вход: Значение element, которое нужно добавить в конец массива.**

**Предусловия: Нет.**

**Процесс: Добавляет элемент в конец массива.**

**Выход: Нет.**

**Постусловия: Размер массива увеличен на 1, элемент element добавлен в конец массива.**

**getData() const:**

**Вход: Нет аргументов.**

**Предусловия: Нет.**

**Процесс: Возвращает указатель на данные массива.**

**Выход: Указатель на данные массива.**

**Постусловия: Нет.**

**pop\_back():**

**Вход: Нет аргументов.**

**Предусловия: Массив не пуст.**

**Процесс: Удаляет последний элемент из массива.**

**Выход: Нет.**

**Постусловия: Размер массива уменьшен на 1.**

**at(size\_t index) const:**

**Вход: Индекс index элемента, который нужно просмотреть.**

**Предусловия: Индекс index находится в допустимых границах массива.**

**Процесс: Возвращает ссылку на элемент массива, находящийся по указанному индексу index.**

**Выход: Ссылка на элемент массива.**

**Постусловия: Нет.**

**operator[](size\_t index) const:**

**Вход: Индекс index элемента, который нужно прочитать.**

**Предусловия: Индекс index находится в допустимых границах массива.**

**Процесс: Возвращает константную ссылку на элемент массива, находящийся по указанному индексу index.**

**Выход: Константная ссылка на элемент массива.**

**Постусловия: Нет.**

**operator[](size\_t index):**

**Вход: Индекс index элемента, который нужно изменить.**

**Предусловия: Индекс index находится в допустимых границах массива.**

**Процесс: Возвращает ссылку на элемент массива, находящийся по указанному индексу index, для его изменения.**

**Выход: Ссылка на элемент массива.**

**Постусловия: Нет.**

**DynamicArray(const DynamicArray& other):**

**Вход: Ссылка на объект DynamicArray other, из которого нужно скопировать данные.**

**Предусловия: Нет.**

**Процесс: Конструктор копирования, создает новый объект DynamicArray и копирует данные из массива other в новый объект.**

**Выход: Новый объект DynamicArray.**

**Постусловия: Создан новый объект DynamicArray с копией данных из объекта other.**

**operator=(const DynamicArray& other):**

**Вход: Ссылка на объект DynamicArray other, из которого нужно скопировать данные.**

**Предусловия: Нет.**

**Процесс: Оператор присваивания копирования, копирует данные из массива other в текущий объект DynamicArray.**

**Выход: Ссылка на текущий объект DynamicArray.**

**Постусловия: Данные из объекта other скопированы в текущий объект DynamicArray.**

**DynamicArray(DynamicArray&& other) noexcept:**

**Вход: Ссылка на объект DynamicArray other, из которого нужно переместить данные.**

**Предусловия: Нет.**

**Процесс: Конструктор перемещения, перемещает ресурсы (данные, емкость и размер) из объекта other в новый объект DynamicArray.**

**Выход: Новый объект DynamicArray.**

**Постусловия: Создан новый объект DynamicArray с перемещенными данными из объекта other.**

**operator=(DynamicArray&& other) noexcept:**

**Вход: Ссылка на объект DynamicArray other, из которого нужно переместить данные.**

**Предусловия: Нет.**

**Процесс: Оператор перемещения, перемещает ресурсы (данные, емкость и размер) из объекта other в текущий объект DynamicArray.**

**Выход: Ссылка на текущий объект DynamicArray.**

**Постусловия: Данные из объекта other перемещены в текущий объект DynamicArray.**

**push\_back\_n(const T& element, size\_t n):**

**Вход: Значение element, которое нужно добавить в конец массива, и количество элементов n.**

**Предусловия: Нет.**

**Процесс: Добавляет n элементов со значением element в конец массива.**

**Выход: Нет.**

**Постусловия: Размер массива увеличен на n, элементы element добавлены**

**resize(size\_t newSize):**

**Вход: Новый размер массива newSize.**

**Предусловия: Новый размер массива может быть больше, меньше или равен текущему размеру.**

**Процесс: Изменяет размер массива до указанного значения newSize. При увеличении размера массива выделяет дополнительную память, если необходимо. При уменьшении размера массива уменьшает его емкость.**

**Выход: Нет.**

**Постусловия: Размер массива изменен до указанного значения newSize.**

**insert\_n(size\_t index, const T& element, size\_t n):**

**Вход: Позиция index, по которой нужно вставить элементы, значение element, которое нужно вставить, и количество элементов n.**

**Предусловия: Нет.**

**Процесс: Вставляет n элементов со значением element в массив, начиная с указанной позиции index. При необходимости увеличивает емкость массива.**

**Выход: Нет.**

**Постусловия: Размер массива увеличен на n, элементы element вставлены в массив.**

**erase\_n(size\_t index, size\_t n):**

**Вход: Позиция index, с которой нужно удалить элементы, и количество элементов n.**

**Предусловия: Нет.**

**Процесс: Удаляет n элементов из массива, начиная с указанной позиции index. При необходимости уменьшает емкость массива.**

**Выход: Нет.**

**Постусловия: Размер массива уменьшен на n.**

**fill\_random(T min, T max):**

**Вход: Минимальное значение min и максимальное значение max для заполнения массива случайными числами.**

**Предусловия: min должно быть меньше max.**

**Процесс: Заполняет массив случайными числами в диапазоне от min до max.**

**Выход: Нет.**

**Постусловия: Массив заполнен случайными числами в указанном диапазоне.**

**seek\_at\_position(size\_t n, const int\* arr, size\_t pos, int x):**

**Вход: Размер массива n, указатель на массив arr, позиция pos, на которой нужно искать элемент, и значение x, с которым сравнивается элемент массива.**

**Предусловия: Нет.**

**Процесс: Ищет элемент x в массиве arr, начиная с указанной позиции pos. Возвращает позицию элемента в массиве, если он найден, или -1, если элемент не найден.**

**Выход: Позиция элемента в массиве или -1, если элемент не найден.**

**Постусловия: Нет.**

**concatenate(const DynamicArray& other):**

**Вход: Ссылка на динамический массив other.**

**Предусловия: Нет.**

**Процесс: Конкатенирует текущий массив с массивом other, добавляя все элементы массива other в конец текущего массива. При необходимости увеличивает емкость текущего массива.**

**Выход: Нет.**

**Постусловия: Размер текущего массива увеличен на количество элементов в массиве other, все элементы массива other добавлены в конец текущего массива.**

**reserve(size\_t newCapacity):**

**Вход: Новая емкость newCapacity для массива.**

**Предусловия: Новая емкость должна быть больше текущей емкости массива.**

**Процесс: Выделяет память под массив с новой емкостью newCapacity. Копирует существующие элементы из старого массива в новый массив. Обновляет емкость текущего массива.**

**Выход: Нет.**

**Постусловия: Емкость текущего массива установлена в newCapacity.**