Домашнее задание № 7.4

Курс:
«Язык программирования Java»

ТЕМА: GENERICS

Написать класс-контейнер, реализующий логику работы одномерного безразмерного динамического массива.
Имя класса *MyArrayList*.
Структура данных – массив.
Типы хранимых данных – объекты любых классов (использовать шаблонное программирование).
Все открытые методы должны содержать *javadoc*.
Все открытые методы должны быть протестированы.
В классе должно быть три поля:

- ссылка на инкапсулированный массив (можно назвать *data*);
- текущее количество существующих элементов в массиве (*int size*);
- текущая емкость массива (*int capacity*, по умолчанию 10).

Задание 1

Создать два конструктора.

- С параметром типа *int*. Задающего начальную емкость массива. Принимает один параметр (задает *capacity*), выделяет память под массив (*size* = 0).
- По умолчанию (без параметров). Который выделяет память под массив на 10 элементов, равных нулю (*capacity* = 10, *size* = 0). Переиспользовать конструктор с параметрами для уменьшения кода.

Задание 2

Реализовать методы:

- геттеры для *size*. Сеттера для *size* не должно быть!
- переопределить метод *toString* и реализовать строковое представление элементов массива через пробел.
- *ensureCapacity* – **закрытый метод**! проверяет, достаточно ли резерва памяти для хранения указанного в параметре количества элементов. Если значение параметра меньше текущего *capacity*, то ничего не происходит. Если значение параметра больше текущего *capacity*, то массив пересоздается, памяти выделяется в 1,5 раза + 1 элемент больше. Существующие элементы переносятся в новый массив. Существующие элементы не должны быть потеряны.

Задание 3

Реализовать методы:

- *pushBack* (добавление элемента в конец массива. Должна быть проверка, достаточно ли памяти! Если памяти не достаточно увеличить емкость массива данных);
- *popFront* (удаление первого элемента из массива);
- *pushFront* (добавление нового элемента в начало массива);
- *insert* (вставка нового элемента в массив по указанному индексу, с проверкой на выход за пределы массива);
- *removeAt* (удаление одного элемента по указанному индексу. Должна быть проверка на допустимость индекса);
- *remove* (удаление одного элемента, значение которого совпадает со значением переданного параметра);

- *removeAll* (удаление всех элементов, значения которых совпадает со значением переданного параметра);
- *popBack* (удаление последнего элемента из массива);
- *clear* (обнуление массива – всем элементам массива по индексам от 0 до *size*-1 присвоить значение *null*, полю *size* присвоить значение 0).

Задание 4

Реализовать методы:

- *isEmpty* (метод возвращает *true*, если *size* = 0, и *false* в обратном случае);
- *trimToSize* (метод подгоняет значение *capacity* под *size*, естественно с перевыделением памяти);
- *indexOf* (линейный поиск слева направо первого вхождения в массив указанного значения. В результате работы вернуть индекс найденного элемента, а *е*сли ничего не найдено, вернуть -1);
- *lastIndexOf* (линейный поиск справа налево вхождения в массив указанного значения. В результате работы вернуть индекс найденного элемента, а *е*сли ничего не найдено, вернуть -1).

Задание 5

Реализовать методы:

- *reverse* (изменение порядка следования элементов в массиве на противоположный);
- *shuffle* (случайное перемешивание элементов массива).

Задание 6

Реализовать методы:

- *equals* (в качестве параметра передается ссылка на другой объект класса *MyArrayList*. Метод сравнивает массивы не только по количеству элементов, но и по их содержимому);
- *getElementAt* (возврат копии элемента массива по указанному индексу, с проверкой на выход за пределы массива);
- переопределить метод *clone* – метод создает точную копию *MyArrayList* и возвращает ссылку на эту копию (неглубокое копирование).