**Разработанная программа должна удовлетворять следующим требованиям:**

* Класс, коллекцией экземпляров которого управляет программа, должен реализовывать сортировку по умолчанию. +
* Все требования к полям класса (указанные в виде комментариев) должны быть выполнены. +
* Для хранения необходимо использовать коллекцию типа java.util.Stack +
* При запуске приложения коллекция должна автоматически заполняться значениями из файла. -
* Имя файла должно передаваться программе с помощью: **аргумент командной строки**. -
* Данные должны храниться в файле в формате json +
* Чтение данных из файла необходимо реализовать с помощью класса java.io.BufferedInputStream -
* Запись данных в файл необходимо реализовать с помощью класса java.io.PrintWriter -
* Все классы в программе должны быть задокументированы в формате javadoc. +
* Программа должна корректно работать с неправильными данными (ошибки пользовательского ввода, отсутсвие прав доступа к файлу и т.п.). +

**В интерактивном режиме программа должна поддерживать выполнение следующих команд:**

* help : вывести справку по доступным командам +
* info : вывести в стандартный поток вывода информацию о коллекции (тип, дата инициализации, количество элементов и т.д.) +
* show : вывести в стандартный поток вывода все элементы коллекции в строковом представлении +
* add {element} : добавить новый элемент в коллекцию +
* update id {element} : обновить значение элемента коллекции, id которого равен заданному+
* remove\_by\_id id : удалить элемент из коллекции по его id +
* clear : очистить коллекцию +
* save : сохранить коллекцию в файл -
* execute\_script file\_name : считать и исполнить скрипт из указанного файла. В скрипте содержатся команды в таком же виде, в котором их вводит пользователь в интерактивном режиме. **+**
* exit : завершить программу (без сохранения в файл) +
* remove\_at index : удалить элемент, находящийся в заданной позиции коллекции (index) +
* remove\_last : удалить последний элемент из коллекции +
* history : вывести последние 7 команд (без их аргументов) +
* min\_by\_transport : вывести любой объект из коллекции, значение поля transport которого является минимальным +
* count\_greater\_than\_view view : вывести количество элементов, значение поля view которых больше заданного +
* print\_ascending : вывести элементы коллекции в порядке возрастания +

**Формат ввода команд:**

* Все аргументы команды, являющиеся стандартными типами данных **+** (примитивные типы, классы-оболочки, String, классы для хранения дат), должны вводиться в той же строке, что и имя команды. **+**
* Все составные типы данных (объекты классов, хранящиеся в коллекции) должны вводиться по одному полю в строку. **+**
* При вводе составных типов данных пользователю должно показываться приглашение к вводу, содержащее имя поля (например, "Введите дату рождения:") **+**
* Если поле является enum'ом, то вводится имя одной из его констант (при этом список констант должен быть предварительно выведен). **+**
* При некорректном пользовательском вводе (введена строка, не являющаяся именем константы в enum'е; введена строка вместо числа; введённое число не входит в указанные границы и т.п.) должно быть показано сообщение об ошибке и предложено повторить ввод поля. **+**
* Для ввода значений null использовать пустую строку. **+**
* Поля с комментарием "Значение этого поля должно генерироваться автоматически" не должны вводиться пользователем вручную при добавлении. **+**

**Описание хранимых в коллекции классов:**

public class Flat { **+**

private Long id; //Поле не может быть null, Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение этого поля должно генерироваться автоматически

private String name; //Поле не может быть null, Строка не может быть пустой

private Coordinates coordinates; //Поле не может быть null

private java.time.ZonedDateTime creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически

private Float area; //Поле не может быть null, Значение поля должно быть больше 0

private int numberOfRooms; //Значение поля должно быть больше 0

private Furnish furnish; //Поле не может быть null

private View view; //Поле не может быть null

private Transport transport; //Поле не может быть null

private House house; //Поле может быть null

}

public class Coordinates { **+**

private Long x; //Поле не может быть null

private double y; //Максимальное значение поля: 960

}

public class House { **+**

private String name; //Поле может быть null

private long year; //Значение поля должно быть больше 0

private long numberOfFloors; //Значение поля должно быть больше 0

}

public enum Furnish { **+**

DESIGNER,

BAD,

LITTLE;

}

public enum View { **+**

YARD,

BAD,

NORMAL,

GOOD;

}

public enum Transport { **+**

NONE,

LITTLE,

ENOUGH;

}