**ДЗ - Структуры данных**

* 🔥**Коллекции и структуры данных** (UML диаграммы) <https://www.programiz.com/dsa/stack>
  + Массив
  + Очередь
  + Стек
  + Список
  + Связанный список
  + Мап
  + Хеш Мап
  + Деревья (двоичные деревья)
  + Граф
  + Куча
  + Дек

**Задание**

1. Изучить все базовые типы данных JavaScript/TypeScript и привести примеры их использования

2. Провести исследование и изучить сложные типы данных (структуры данных и коллекции). Привести примеры их использования и объяснить как они работают.. ПРИМЕР: List(ArrayList, LinkedList), Set(HashSet, TreeSet), Map(HashMap, TreeMap), Stack, Queue и т.п.

**Задачи:**

**Задача 1**: Предположим, что Вы разрабатываете программу для учета автомобилей на парковке. По требованию заказчика нужно будет сохранять гос номера автомобиля в виде списка. Удалять из списка автомобили не нужно, только добавлять их в конец списка. При этом допускается, что гос номера могут повторяться.

Напишите программу, которая во время работы будет ожидать ввода в консоли нового гос номера автомобиля и после ввода добавлять его в список. При вводе слова "СПИСОК" будет выводить весь список гос номеров автомобилей и дальше ожидать команды. При вводе слова “СТОП” будет заканчивать работу. Используйте для решения задачи коллекции. Выберите ту коллекцию, которая будет удовлетворять требованиям заказчика и при этом будет максимально продуктивной.

**Задача 2**: Заказчик из прошлого задания решил обновить требования к программе. Теперь нужно будет иметь возможность вставлять новые автомобили не только в конец списка, но и в начало или в середину.

Напишите программу, которая во время работы будет ожидать ввода в консоли нового гос номера автомобиля, затем номер по которому этот автомобиль должен быть в списке и после ввода добавлять его в список. Если номер указан больше, чем в списке автомобилей, тогда добавить запись в конец списка. При вводе слова "СПИСОК" будет выводить весь список гос номеров автомобилей и дальше ожидать команды. При вводе слова “СТОП” будет заканчивать работу. Используйте для решения задачи коллекции. Выберите ту коллекцию, которая будет удовлетворять требованиям заказчика и при этом будет максимально продуктивной.

**Задача 3**: Заказчик снова изменил требования, теперь ему не важен порядок в списке автомобилей, но ему важно, чтоб гос номера автомобилей не повторялись.

Напишите программу, которая во время работы будет ожидать ввода в консоли нового гос номера автомобиля и после ввода добавлять его в список. При вводе слова "СПИСОК" будет выводить весь список гос номеров автомобилей и дальше ожидать команды. При вводе слова “СТОП” будет заканчивать работу. Выберите ту коллекцию, которая будет удовлетворять требованиям заказчика и при этом будет максимально продуктивной.

**Задача 4**: И снова заказчик решил изменить работу программы. На этот раз он хочет, чтобы в списке гос номеров автомобилей были и их владельцы. По гос номеру автомобиля можно будет узнать его владельца. За одним автомобилем может быть закреплен только один владелец, но один владелец может быть закреплен за многими автомобилями. В случае внесения в список гос номера, который уже был внесен ранее, нужно обновлять его владельца.

Напишите программу, которая во время работы будет ожидать ввода в консоли нового гос номера автомобиля, а также его владельца и после ввода добавлять их в список. При вводе слова "СПИСОК" будет выводить весь список гос номеров автомобилей и дальше ожидать команды. При вводе слова “СТОП” будет заканчивать работу. При вводе слова "АВТОМОБИЛИ" будет выводить весь список гос номеров автомобилей При вводе слова "ВЛАДЕЛЬЦЫ" будет выводить весь список владельцев автомобилей. Используйте для решения задачи коллекции. Выберите ту коллекцию, которая будет удовлетворять требованиям заказчика и при этом будет максимально продуктивной.

**Задача 5**: Проверить скорость работы основных коллекций интерфейса List(ArrayList, LinkedList). Создайте экземпляры этих коллекций с 1 000 000 элементов. Заполните их случайными числами. Проверьте скорость работы основных операций каждой коллекции: добавление в конец списка, добавление в середину списка, получение элемента из конца списка, получение элемента из середины списка, удаление элемента из конца списка, удаление элемента из середины списка.

**Задача 6**: Создать класс, который будет хранить в себе данные пользователя, а именно: персональный номер, имя, фамилия, возраст. Создать коллекцию таких пользователей и добавить туда 10 элементов. Отсортировать коллекцию по персональному номеру по возрастанию используя Comparable и Comparator. Вывести коллекцию до и после сортировки.

**Задача 7**: Создать класс сортировок, в котором создать 5 статических методов, каждый из которых должен сортировать массив своим способом. То есть 5 методов - 5 способов сортировки. Каждый метод принимает в качестве параметра - массив чисел и возвращает его отсортированный, по возрастанию, вариант. В конце работы метода вывести время, за которое массив был отсортирован. Методы сортировки не просто копировать, а понимать и уметь объяснить как каждый из них работает.

Программный код должен быть максимально понятным(чистым), используйте комментарии для описания методов сортировки и указания его алгоритмической сложности.