Коллекции объектов

- Есть несколько способов хранения объектов
- Массив - самый эффективный но имеет ограниченный размер
- Классы контейнеров - List,Set,Queue,Map - способны автоматически изменяться в размерах
- В контейнерах нельзя хранить примитивы

- Обобщенные типы и классы, безопасные по отношению к типам
- До контейнеров проблема - компилятор позволяет вставлять объекты неправильного типа, происходила ошибка во время выполнения
- Использование обобщенных типов - ArrayList<type> -> при выборке данных не нужно преобразование
- + можно помещать разные объекты используя восходящее преобразование
- Основные концепции
- Коллекция - последовательность отдельных элементов, формируется по некоторым правилам List - элементы в опр последовательности, Set - хранит только уникальные значения, Queue - элемент выдает элементы в определенном порядке
- Карта - набор пар ключ-значения с выборкой по ключу.
- Для перебора всех контейнеров может применяться синтаксис foreach/итераторы

- Добавление групп элементов
- В коллекциях и массивах
- Collection.addAll() - только другой объект Collection. Array.asList() можно использовать как List, но представление данных как массив -> фиксированный размер
- Подсказка с явно указанным аргументом типа в случае восходящего преобразования
- List<type> myList = Arrays.<type>asLst(добав арг)
- Вывод контейнеров
- Выводятся нормально
- Set - самая быстрая выборка элементов
- Map - нахождение значения по ключу
- TreeMap – ключи по возрастанию, LinkedHashMap – ключи в порядке вставки

- List
- Гарантирует хранения списка в определенной последовательности
- ArrayList - высокая скорость произвольного доступа, медленная вставка\удаление
- Linkedlist - быстрые операции вставки удаления
- Cointains - содержит ли, remove - удаление, indexof - индекс ссылки
- RetreinAll - копирование с пересечением

- Итераторы
- Позволяют работать с контейнером, вне зависимости от его типа
- Итератор - объект, обеспечивающий перемещение по последовательности объектов с выбором каждого объекта + легко весный объект -> существуют ограничения
- Итераторы - может вернуть начальный элемент, получить следующий элемент, проверить есть ли объекты в послдеовательности, удалить из последовательности послдений элемент
- + Операция перебора элементов послдеовательности не зависит от структуры последовательности
- ListIterator - работает только с list, позволяет установить позицию, с которой начинается перебор

- LinkedList - выполняет операции вставки и удаления более эффективно
- Реализует базовый интерфейс, есть методы, позваляющие работать как со стеком, очередью, двухсторонней очередью
- Стек - первый вошел, последний вышел, push - поместили в последнюю очередь, pop - вытолкнули и удалили, peek - просто вытолкнули. LinkedList - более качественная реализация стека

- Set - содержит не более одного экземпляра каждого объекта. Нет дополнительной функциональности, проверяет присутствие объекта по значению объекта
- HashSet - использует хэширование для скорости -> нет определенного порядка
- + TreeSet - структура красно-черное дерево -> тоже нет определенного порядка

- Map - ключ\значение , чтоб по одному ключу группа значений можно - Map<type,Collection<type>> Может вернуть множество своих ключей, коллекцию своих значений множество пар, keySet возвращает контейнер Set содержащий все ключи из petPeople

- Очередь
- Контейнер FIFO - порядок занесение == порядок извлечения
- Offer - вставляет элемент в конец очереди или false
- Peek/element - возвращают елемент в начале очереди без его извлечения, но peek – null для пустой очереди, element - исключение
- Poll remove - извлечение и возвращение элемента - poll – null? Remove - исключение в случае ошибки
- PriorityQueue - описывает стандартные правила для определения следубщего элемента - следующий - элемент, обладающий наивысшим приоритетом
- Создание приоритетной очереди - собственноручно

- Collection - интерфейс, описывающий все последовательные контейнеры \
- Позволяет создать более универсальный код, тк код для интерфейса может применяться к большему количеству объектов
- Реализаия Collection означает поддержку итератора

- Foreach и итераторы
- Форич - работает для любой коллекции, тк любой класс, реализующий Iterable можно использовать с форич

- Метод-Адаптер
- Добавления новых способов использования foreach - реализация метода, создающего объект Iterable - внутри Iterator

- Массив - объект - числовой индекс, известный тип, способен хранить примитивы, фикс размер
- Коллекция - одиночные элементы, карта - ключ\значение. Используются обобщенные типы, указывается что именно. Изменяют свои размеры. НЕ может хранить примитивы, но может хранить обертки
- Лист - ассоциирует с объектами числовые индексы -> упорядоченный контейнер
- ArrayList - получение произвольного элемента, LinkedList - множество вставок и удалений в середине списка(можно использовать как очередь и стек)
- Карта - объект\объект, HashMap - быстрый доступ к элементам, TreeMap - ключи сортированы. LinkedHashMap - ключи сортированы в порядке добавления
- Set - уникальные элементы. HashSet - максимально ускоренный поиск, TreeSet - отсортированные элементы. LinkedHashSet - эементы в порядке добавления