Основные методы **Map**:

* **put(K key, V value)** — добавление элемента в Map;
* **get(Object key)** — поиск значения по ключу;
* **containsKey(Object key)** — проверка Map на наличие данного ключа;
* **containsValue(Object value)** — проверка Map на наличие данного значения;
* **remove(Object key)** — удаление значения по его ключу.

JDBC используется для подключения к базе данных это как бы драйвера как бывают такие как для компа .

Итератор — объект, позволяющий перебирать элементы коллекции

Конструктор нужен для того чтобы проинициализировать экземпляр класса.

Модификаторы доступа private, public, package private, protected это как прайват только на весь класс.

Виды переменных String , int , Boolean , Double, char.

Типы данных ссылочные и примитивные .

JVM исполняет байт код

JRE – среда выполнения в JRE входит JVM и стандартная библиотека Java этого достаточно чтобы на компе запустить приложение написанное на Java .

JDK – набор для разработчика туда входит все в том числе и компилятор.

Методы класса Object – equals ,hashcode, toString , notify , wait , clone , getClass

Переопределение метода – это переопределение метода который мы унаследовали от родительского класса . Концепция у объекта остается но как он будет её делать это меняется в нашем классе

String immutable он неизменный . Его нельзя изменить после создания.   
Так же String final нельзя создавать подтипы его .

StreamApi нужен чтобы потоково обрабатывать данные в коллекции

 Hibernate — это популярный framework, цель которого связать ООП и реляционную базу данных. Работа с Hibernate сократит время разработки проекта в сравнении с обычным jdbc.

**Какова иерархия коллекций?**

1. **Iterable**
2. **Collection**
3. **List, Queue, Set**

#### В чем отличия/сходства List и Set?

#### Оба унаследованы от Collection, а значит имеют одинаковый набор и сигнатуры методов. List хранит объекты в порядке вставки, элемент можно получить по индексу. Set не может хранить одинаковых элементов.

#### 

|  |  |  |
| --- | --- | --- |
| Интерфейс | Класс/Реализация | Описание |
| List | ArrayList | Список |
| LinkedList | Список |
| Vector | Вектор |
| Stack | Стек |
| Set | HashSet | Множество |
| TreeSet | Множество |
| SortedSet (расширяющий интерфейс) | Отсортированное множество |
| Map | HashMap | Карта/Словарь |
| TreeMap | Карта/Словарь |
| SortedMap (расширяющий интерфейс) | Отсортированный словарь |
| Hashtable | Хеш-таблица |

#### Что разного/общего у классов ArrayList и LinkedList, когда лучше использовать ArrayList, а когда LinkedList?

ArrayList реализован внутри в виде обычного массива. Поэтому при вставке элемента в середину, приходится сначала сдвигать на один все элементы после него, а уже затем в освободившееся место вставлять новый элемент. Зато в нем быстро реализованы взятие и изменение элемента – операции get, set, так как в них мы просто обращаемся к соответствующему элементу массива.

LinkedList реализован внутри по-другому. Он реализован в виде связного списка: набора отдельных элементов, каждый из которых хранит ссылку на следующий и предыдущий элементы. Чтобы вставить элемент в середину такого списка, достаточно поменять ссылки его будущих соседей. А вот чтобы получить элемент с номером 130, нужно пройтись последовательно по всем объектам от 0 до 130. Другими словами операции set и get тут реализованы очень медленно.

Если необходимо вставлять (или удалять) в середину коллекции много элементов, то лучше использовать LinkedList. Во всех остальных случаях – ArrayList.

LinkedList требует больше памяти для хранения такого же количества элементов, потому что кроме самого элемента хранятся еще указатели на следующий и предыдущий элементы списка, тогда как в ArrayList элементы просто идут по порядку.

122) Чем отличаются HashMap и TreeMap?

HashSet реализован на основе хеш-таблицы, а TreeSet — на основе бинарного дерева.

HashSet гораздо быстрее чем TreeSet (константное время против логарифмического для большинства операций, таких как add, remove, contains), но TreeSet гарантирует упорядоченность объектов. Оба не синхронизированы.

TreeSet гарантирует порядок элементов

TreeSet не предоставляет каких-либо параметров для настройки производительности

11)Что такое Hashtable, чем она отличается от HashMap?

Некоторые методы HashTable синхронизированы, поэтому она медленнее HashMap.

HashTable синхронизирована, а HashMap нет.

HashTable не позволяет иметь null ключи или значения. HashMap позволяет иметь один null ключ и сколько угодно null значений.

SQL

**INSERT** – это вид запроса SQL, при применении которого СУБД выполняет добавление новых записей в БД.

### **SELECT – оператор запроса возвращает набор данных**

### **DROP** – ключевое слово в SQL, применяемое для удаления данных с помощью запроса. К примеру удаление некоторой таблицы из БД.

**SELECT** ('столбцы или \* для выбора всех столбцов; обязательно')

**FROM** ('таблица; обязательно')

**WHERE** ('условие/фильтрация, например, city = 'Moscow'; необязательно')

**GROUP** **BY** ('столбец, по которому хотим сгруппировать данные; необязательно')

**HAVING** ('условие/фильтрация на уровне сгруппированных данных; необязательно')

**ORDER** **BY** ('столбец, по которому хотим отсортировать вывод; необязательно')

JOIN — необязательный элемент, используется для объединения таблиц по ключу, который присутствует в обеих таблицах. Перед ключом ставится оператор ON

**select** \* **from** Orders

**JOIN** Customers **ON** Orders.CustomerID = Customers.CustomerID