1. What is a Map in Java?

   A Map in Java is an interface that represents a collection of key-value pairs where each key is unique, and each key maps to exactly one value. It is a part of the java.util package and is widely used for data storage and retrieval. Keys are used to retrieve values, and no duplicate keys are allowed.

2. What are the commonly used implementations of Map in Java?

- HashMap: A hash table-based implementation of Map. It allows null keys and values and provides average constant-time performance for operations like get and put.
- TreeMap: A Map implementation that maintains keys in a sorted order. It does not allow null keys but allows null values.
- LinkedHashMap: Maintains a predictable iteration order, either based on insertion order or access order.

3. What is the difference between HashMap and TreeMap?

   hashmap
   - No order (unordered collection).
   - Allows one null key and multiple null values.
   - O(1) average for get/put operations.

   Treemap
   - Keys are sorted in natural or custom order.
   - Does not allow null keys but allows null values.
   - O(log n) for get/put operations.

4. How do you check if a key exists in a Map in Java?

   We have to use the containsKey() method

5. What are Generics in Java?

Generics are a feature in Java that allows types (classes and interfaces) to be parameterized. Generics enable code reusability and type safety by ensuring that only specific types of objects are used in a collection or class at compile time.

6. What are the benefits of using Generics in Java?

   - Type Safety: Ensures that a collection or class contains only specific types of objects.
   - Eliminates Casting: Reduces the need for explicit casting, leading to cleaner code.
   - Compile-Time Checking: Detects errors at compile time rather than runtime.
   - Code Reusability: Allows for the creation of reusable code that can handle different types.

7. What is a Generic Class in Java?

   A generic class is a class that can operate on any specified type. The type is specified at the time of instantiation using type parameters. Ex-

```
public class Box<T> {
   private T item;
   public void setItem(T item) { this.item = item; }
   public T getItem() { return item; }
}
Box<String> box = new Box<>();
box.setItem("Hello");
```

8. What is a Type Parameter in Java Generics?

   A type parameter is a placeholder for a type that is provided when a generic class, interface, or method is instantiated or invoked. Commonly used type parameters:

   - T – Type
   - E – Element
   - K – Key

- V – Value

9. What is a Generic Method in Java?

   A generic method is a method that declares its own type parameter, which is used within the method. It can be defined in a generic or non-generic class.

```
public static <T> void printArray(T[] array) {
   for (T element : array) {
      System.out.println(element);
   }
}
```

10. What is the difference between ArrayList and ArrayList<T>?

    ArrayList: A raw type that does not enforce type safety. Any type of object can be added, leading to potential runtime ClassCastException.
    ArrayList<T>: A generic version of ArrayList that specifies the type of elements it holds, ensuring type safety.

```
ArrayList list = new ArrayList();        // Raw type
list.add("String");
list.add(10);                            // Compiles but may cause issues.

ArrayList<String> listT = new ArrayList<>();  // Generic type
listT.add("String");
// listT.add(10);                        // Compile-time error
```