Collections in Java

The Backbone of Test Frameworks

Key collection interfaces?

- List - Ordered collection, allows duplicates
- Set - Unordered, no duplicates
- Map - Key-value pairs
- Queue - FIFO structure

Common classes used in automation?

- Array List - Store Web Elements, input sets
- HashMap - Hold config, headers, key-values
- HashSet - Validate uniqueness (e.g., dropdowns)
- LinkedList - Used in queue-based flows

Why Collections/Arrays utils?

- Collections: sorting, thread-safe wrappers, min/max
- Arrays: array-to-list conversion, sorting primitives

www.linkedin.com/in/vishupriyaravichandran

Selenium Automation Use Cases

Why prefer ArrayList in Selenium?

- Works with findElements(), stores dynamic WebElements
- Fast index-based access for loops

How to store elements from findElements()?

- Use List<WebElement> to loop through elements

```
List<WebElement> elements = driver.findElements(By.tagName("button"));
for (WebElement btn : elements) {
    System.out.println(btn.getText());
}
```

How to map test steps or locators?

- Use Map<String, By> to map locator keys to locators

Where is Set used in Selenium?

- For window handles
- To validate uniqueness in dropdowns

How to manage test data in DDT?

- Map<String, String> for a single test row

- List<Map<String, String>> for all test rows

How to remove duplicates from test data?

- Store values in a Set to auto-remove duplicates

How to remove duplicates from a List?

```
List<String> unique = new ArrayList<>(new HashSet<>(originalList));
```

How to maintain insertion order in tests?

- Use LinkedHashSet or LinkedHashMap

When to use Deque in browser tests?

- Simulate browser back/forward actions

```
Deque<String> navStack = new ArrayDeque<>();
```

API Testing with Collections

How to pass headers in API calls?

- Use Map<String, String> to construct headers
- Easily reusable across requests (auth, content-type)
- Works well with REST Assured .headers() method

```
Map<String, String> headers = new HashMap<>();
headers.put("Authorization", "Bearer token");
headers.put("Content-Type", "application/json");
```

How to build dynamic JSON payloads?

- Use Map<String, Object> to simulate JSON object
- Useful for user creation, updates, login payloads
- Allows flexible key-value insertions based on test case

```
Map<String, Object> user = new HashMap<>();
user.put("id", 101);
user.put("name", "Priya");
```

How to represent a JSON array of objects?

- *Use List<Map<String, Object>> for multiple user-like objects*
- *Simulates payloads like [{...}, {...}]*
- *Ideal for bulk creation APIs or multi-record validation*

```
List<Map<String, Object>> users = new ArrayList<>();users.add(user);
```

How to extract a list from API response?

- *Use .getList() with JsonPath for list values*
- *Handy for validating response IDs, emails, names*
- *Supports direct assertions or further processing*

```
List<String> ids = response.jsonPath().getList("users.id");
```

How to validate uniqueness in API results?

- *Convert list to Set and compare size*

- *Quick way to detect duplicates in ID, email, etc.*
- *Works well in response validations or assertions*

```
Set<String> unique = new HashSet<>(emails);
assert emails.size() == unique.size();
```

How to handle multiple API responses?

- *Store each response in a Map<String, Response>*
- *Label by test step or API endpoint*
- *Helps when chaining token generation → user access → validations*

Data Mapping & Configuration

How to store config or environment data?

```
Map<String, String> config = new HashMap<>();
config.put("baseUrl", "https://api.example.com");
```

Where are Maps commonly used in frameworks?

- Step names, locator mapping, config, headers

How to convert between List, Set, and Map?

```
Set<T> s = new HashSet<>(list);
List<T> l = new ArrayList<>(set);
List<K> keys = new ArrayList<>(map.keySet());
```

How to iterate a Map with key-value pairs?

```
for (Map.Entry<String, String> entry : map.entrySet()) {
    System.out.println(entry.getKey() + ": " + entry.getValue());
}
```

When to use TreeMap or TreeSet?

- TreeMap - When keys need to be sorted
- TreeSet - When values must be unique and sorted

How HashMap differs from LinkedHashMap?

- HashMap - Fast, no order guarantee
- LinkedHashMap - Maintains insertion order

Thread Safety & Concurrency

How to make Collections thread-safe?

- Use Collections.synchronizedList()
- Use thread-safe collections for multi-threaded tests

When to use ConcurrentHashMap in tests?

- When sharing data like tokens across threads

How to use CopyOnWriteArrayList in tests?

- Thread-safe alternative to ArrayList
- Good for read-heavy test scenarios
- Safe to iterate even when modified by other threads

```
List<String> list = new CopyOnWriteArrayList<>();
list.add("email1@example.com");
list.add("email2@example.com");

for (String email : list) {    System.out.println(email);
```