**National University of Computer and Emerging Sciences **

**Laboratory Manual**

*for*

**Data Structures Lab**

| Course Instructor | Sir Waqas Ali |
| --- | --- |
| Lab Instructor(s) | Zain Ali Nasir, Hira Tayyab |
| Section | BSE 5C & B |
| Date | Wednesday, Sep 25, 2024 |
| Semester | Fall 2024 |

**Department of Software Engineering**

FAST-NU, Lahore, Pakistan

**In this lab we cover:**

* Generics and Collections

**What are Generics in Java?**

Generics in Java allow for type-safe code by enabling classes, methods, and interfaces to operate on any specified data type without needing to cast or use Object type. The key concept behind generics is type parameterization, which means you can define a class or method that can work with any type, while still ensuring type safety.

#### Key Benefits of Generics:

1. **Type Safety:** Generics enforce compile-time type checking, reducing runtime errors like ClassCastException. For example, using a List<Integer> ensures that only integers can be added.
2. **Reusability:** You can write a single generic method or class that works with different types, eliminating the need for code duplication.
3. **Elimination of Casting:** Since you specify the type, you avoid unnecessary casting of objects

**Example:**

// Generic class

class Box<T> {

private T item;

public void add(T item) {

this.item = item;

}

public T get() {

return item;

}

}

public class Main {

public static void main(String[] args) {

//Same class behave as a integer

Box<Integer> integerBox = new Box<>();

integerBox.add(100);

Integer value = integerBox.get(); // No need for casting

System.out.println(value);

//Same class behave as an String

Box<String> str = new Box<>();

str.add("This is good.");

String value1 = str.get(); // No need for casting

System.out.println(value1);

}

}

**What are Collections?**

Collections in Java are a set of classes and interfaces that provide data structures like lists, sets, and maps to store and manipulate groups of objects. The Java Collections Framework (JCF) is part of the java.util package and offers various implementations of these data structures.

#### Key Interfaces of Java Collections:

1. **List:** An ordered collection (also known as a sequence) that allows duplicate elements. Implementations include ArrayList, LinkedList, etc.
2. **Set:** A collection that does not allow duplicate elements. Implementations include HashSet, TreeSet, etc.
3. **Map:** A collection that stores key-value pairs, where keys are unique.

import java.util.ArrayList;

import java.util.List;

public class CollectionExample {

public static void main(String[] args) {

// Create a List of Strings

List<String> names = new ArrayList<>();

names.add("BS-SE(5A)");

names.add("BS-SE(6A)");

names.add("BS-SE(7A)");

// Iterate through the List

for (String name : names) {

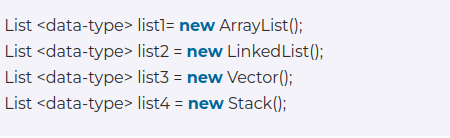
System.out.println(name);

}

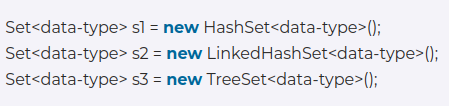
}

}

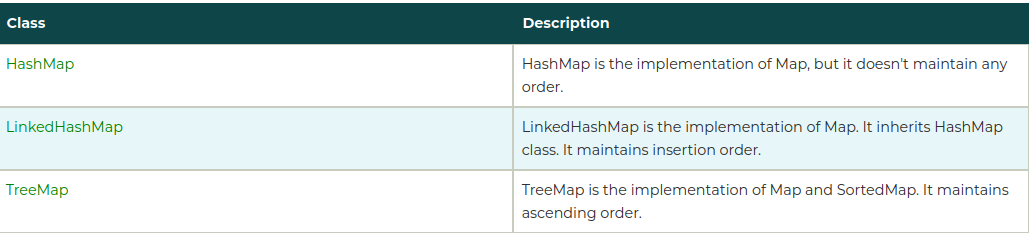
**List Collections**



**Set Collections**



**Map Collections**

****

**Code Example:**

import java.util.HashMap;

import java.util.Map;

public class MapExample {

public static void main(String[] args) {

// Create a HashMap to store students' names and their grades

Map<String, Integer> studentGrades = new HashMap<>();

// Add key-value pairs to the map

studentGrades.put("Ali", 85);

studentGrades.put("Umer", 92);

studentGrades.put("AbuBakar", 78);

studentGrades.put("Usman", 90);

// Print the map

System.out.println("Initial Map: " + studentGrades);

// Access value using a key

int aliceGrade = studentGrades.get("Umer");

System.out.println("Umer's Grade: " + aliceGrade);

// Check if a key exists

if (studentGrades.containsKey("AbuBakar")) {

System.out.println("AbuBakar is in the map.");

}

// Iterate over the map's entries

System.out.println("Student Grades:");

for (Map.Entry<String, Integer> entry : studentGrades.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

// Remove a key-value pair

studentGrades.remove("Usman");

System.out.println("After removing Usman: " + studentGrades);

// Replace a value for an existing key

studentGrades.put("AbuBakar", 95);

System.out.println("After updating David's grade: " + studentGrades);

}

}

**Iterators**

**Iterator** object, which allows a program to walk through the

collection and remove elements from it during the iteration.

**Code Example:**

Iterator< String > iterator = collection1.iterator();

// loop while collection has items

while ( iterator.hasNext() )

{

if ( collection2.contains( iterator.next() ) )

iterator.remove(); // remove current Color

} // end while

**Map Iteration**

**1) entrySet**

Iterator<Map.Entry<Integer, String>> iterator = map.entrySet().iterator(); // Iterate using the Iterator

while (iterator.hasNext())

{

Map.Entry<Integer, String> entry = iterator.next(); System.out.println(entry.getKey() + " " + entry.getValue());

}

**2) KeySet**

Iterator<Integer> iterator = map.keySet().iterator();

while (iterator.hasNext())

{

Integer key = iterator.next();

String value = map.get(key);

System.out.println(key + " " + value);

}

**Lab Manual**

**Problem Statement 1: SEO Analyzer**

Ali is running an SEO agency and he wants to automate page analyzer in java. As a software engineer you need to develop an application *Jsoup,* a Java base library for scraping.

Install jsoup dependency : https://jsoup.org/

String url = "https://en.wikipedia.org/";

try {

// Fetch the HTML content from the URL

Document document = Jsoup.connect(url).get();

Get the page content and sort the tags which contain **{tag: frequency}** and words on which page is SEO optimized like how much time **keywords** exist on this page.

**Requirements:**

1. If more than 5 image tags exist on the page , the system will show a message.
2. Frequency of keywords, keywords entered by the user.
3. Frequency of html CSS tags in sorted order.
4. Users can enter multiple URLs.

Use appropriate collections types for this task.

**Avoid any generative AI helping tool.**