

**Session 3+4****Collections API & Generics****Phần I - Thực hiện trong 90 phút****1. Mục tiêu**

- Hiểu biết về gói java.util
- Nắm vững khái niệm Collection trong Java và ưu điểm của nó so với sử dụng mảng.
- Cách khai báo và sử dụng List, Set, Map.

**2. Thực hiện**

**Bài thực hành 1:** Viết class StudentManager có biến lstStudent kiểu List để nhập, sửa, xóa và hiển thị danh sách sinh viên lớp Java cơ bản.

Bước 1: Tạo class StudentManager.

```
package demo.jp2.lab02.phan2.bai1;

import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author minhvuvc
 */
public class StudentManager {

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

    //Bước 2: Viết hàm nhập dữ liệu.
    public void testList() {
        lstSinhVien.add("Tuấn");
        lstSinhVien.add("Hà");
        lstSinhVien.add("Linh");
        System.out.println(" ArrayList Student");
        System.out.print("\t" + lstSinhVien + "\n");
    }

    //Bước 3: Viết hàm sửa dữ liệu.
    public void updateList() {
        lstSinhVien.set(1, "Hồng Hà");

        System.out.println("Update ArrayList Student");
        System.out.print("\t" + lstSinhVien + "\n");
    }
}
```

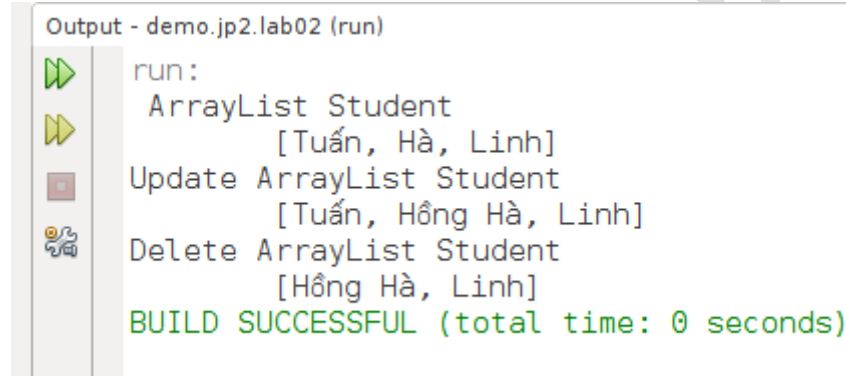
```
//Bước 4: Viết hàm xóa dữ liệu
public void deleteList() {

    lstSinhVien.remove(0);

    System.out.println("Delete ArrayList Student");
    System.out.print("\t" + lstSinhVien + "\n");
}

/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    StudentManager studentManager = new StudentManager();
    studentManager.testList();
    studentManager.updateList();
    studentManager.deleteList();
}
}
```

Bước 5: Chạy chương trình.



```
Output - demo.jp2.lab02 (run)
run:
  ArrayList Student
    [Tuấn, Hà, Linh]
  Update ArrayList Student
    [Tuấn, Hồng Hà, Linh]
  Delete ArrayList Student
    [Hồng Hà, Linh]
BUILD SUCCESSFUL (total time: 0 seconds)
```

**Bài thực hành 2:** Tạo đối tượng Set sử dụng kiểu dữ liệu Integer. Thêm dữ liệu ngẫu nhiên các số nguyên bất kỳ sau đó in từng phần tử trong danh sách ra màn hình.

Bước 1: Tạo class SetDemo.

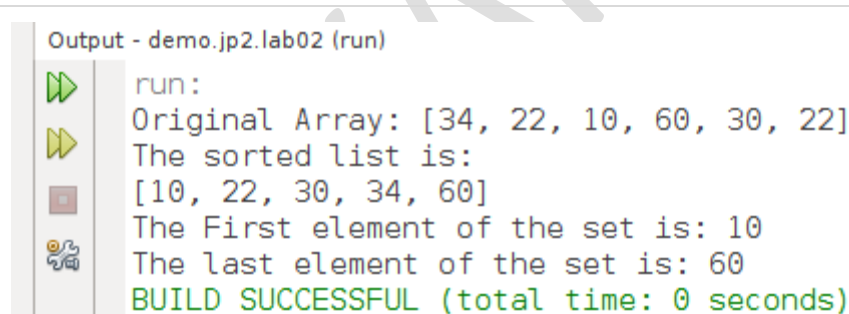
```
package demo.jp2.lab02.phan2.bai2;

import java.util.Arrays;
import java.util.TreeSet;

/**
 *
 * @author minhvuvc
```

```
*/  
public class SetDemo {  
  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
        //Bước 2: Tạo đối tượng Set từ TreeSet và thêm dữ liệu.  
        int count[] = {34, 22, 10, 60, 30, 22};  
        System.out.println("Original Array: " + Arrays.toString(count));  
  
        TreeSet sortedSet = new TreeSet<Integer>();  
        for (int i = 0; i < 5; i++) {  
            sortedSet.add(count[i]);  
        }  
  
        System.out.println("The sorted list is:");  
        System.out.println(sortedSet);  
  
        System.out.println("The First element of the set is: "  
            + (Integer) sortedSet.first());  
        System.out.println("The last element of the set is: "  
            + (Integer) sortedSet.last());  
    }  
}
```

Bước 3: Chạy chương trình.



```
Output - demo.jp2.lab02 (run)  
run:  
Original Array: [34, 22, 10, 60, 30, 22]  
The sorted list is:  
[10, 22, 30, 34, 60]  
The First element of the set is: 10  
The last element of the set is: 60  
BUILD SUCCESSFUL (total time: 0 seconds)
```

**Bài thực hành 3:** Viết một chương trình Java sử dụng Collection Map để lưu trữ dữ liệu mã sinh viên và tên sinh viên.

Bước 1: Tạo lớp MapDemo.

```
package demo.jp2.lab02.phan2.bai3;  
  
import java.util.HashMap;  
import java.util.Map;
```

```
/**
 *
 * @author minhvuvc
 */
public class MapDemo {

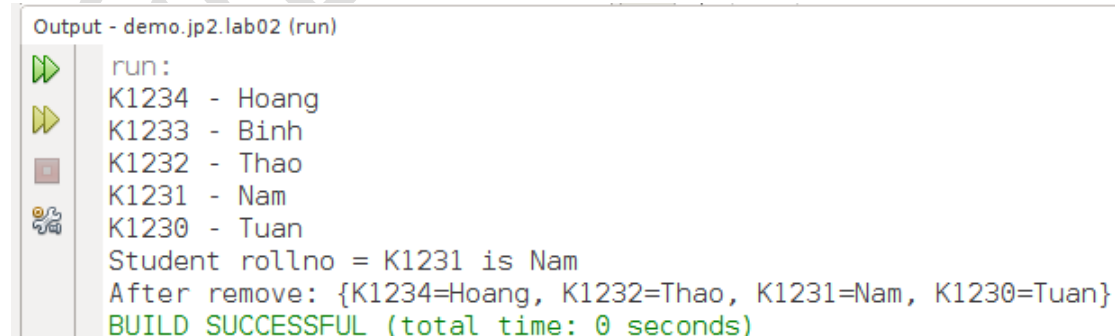
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        //Bước 2: Tạo đối tượng Map và chèn dữ liệu.
        Map<String, String> studentBKAP = new HashMap<>();
        studentBKAP.put("K1230", "Tuan");
        studentBKAP.put("K1231", "Nam");
        studentBKAP.put("K1232", "Thao");
        studentBKAP.put("K1233", "Binh");
        studentBKAP.put("K1234", "Hoang");

        for (Map.Entry<String, String> entrySet : studentBKAP.entrySet()) {
            String key = entrySet.getKey();
            String value = entrySet.getValue();
            System.out.println(key + " - " + value);
        }

        System.out.println("Student rollno = K1231 is " + studentBKAP.get("K1231"));

        // Xoa sinh vien ma K1233
        studentBKAP.remove("K1233");
        System.out.println("After remove: " + studentBKAP);
    }
}
```

Bước 3: Chạy chương trình.



```
Output - demo.jp2.lab02 (run)
run:
K1234 - Hoang
K1233 - Binh
K1232 - Thao
K1231 - Nam
K1230 - Tuan
Student rollno = K1231 is Nam
After remove: {K1234=Hoang, K1232=Thao, K1231=Nam, K1230=Tuan}
BUILD SUCCESSFUL (total time: 0 seconds)
```

## Phần II - Thực hiện trong 30 phút

### 1. Mục tiêu

- Hiểu khái niệm Generics.
- Khai báo và sử dụng những class, method Generics.

### 2. Thực hiện

**Bài thực hành 1:** Viết một class chứa dữ liệu Generics tên là MyGenerics, từ class tên là MainClass gọi và sử dụng.

Bước 1: Tạo class và khai báo biến kiểu T.

```
package demo.jp2.lab02.phan1.bai1;

/**
 *
 * @author minhvuvc
 */
public class MyGenerics<T> {

    private T t;

    //Bước 2: Viết hàm get-set.
    public T getT() {
        return t;
    }

    public void setT(T t) {
        this.t = t;
    }
}
```

Bước 3: Từ main() của MainClass tạo 2 biến MyGenerics với kiểu dữ liệu lần lượt là String và Integer.

```
package demo.jp2.lab02.phan1.bai1;

/**
 *
 * @author minhvuvc
 */
public class MainClass {

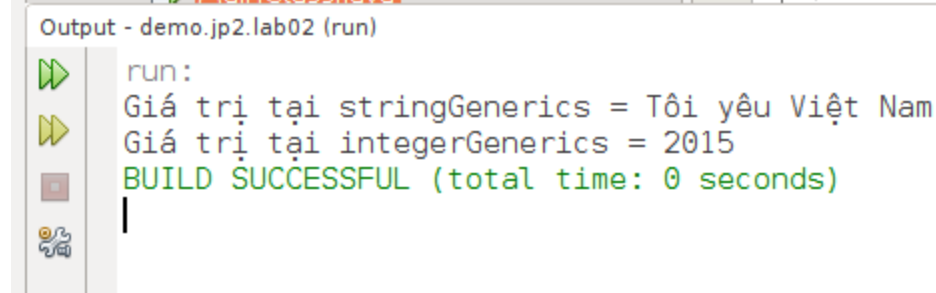
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        MyGenerics<String> stringGenerics = new MyGenerics<String>();
    }
}
```

```
MyGenerics<Integer> integerGenerics = new MyGenerics<Integer>();

//Bước 4: Chèn dữ liệu tương ứng và in dữ liệu ra màn hình.
stringGenerics.setT(new String("Tôi yêu Việt Nam"));
integerGenerics.setT(new Integer(2015));

System.out.println("Giá trị tại stringGenerics = " + stringGenerics.getT());
System.out.println("Giá trị tại integerGenerics = " + integerGenerics.getT());
}
}
```

Bước 5: Chạy và xem kết quả.



```
Output - demo.jp2.lab02 (run)
run:
Giá trị tại stringGenerics = Tôi yêu Việt Nam
Giá trị tại integerGenerics = 2015
BUILD SUCCESSFUL (total time: 0 seconds)
```

**Bài thực hành 2:** Viết một method dạng Generics trong class MainClass có nhiệm vụ in ra danh sách mảng (kiểu dữ liệu bất kỳ). Trong hàm main() gọi và sử dụng.

Bước 1: viết method printArray() trong class MainClass có hàm main().

```
package demo.jp2.lab02.phan1.bai2;

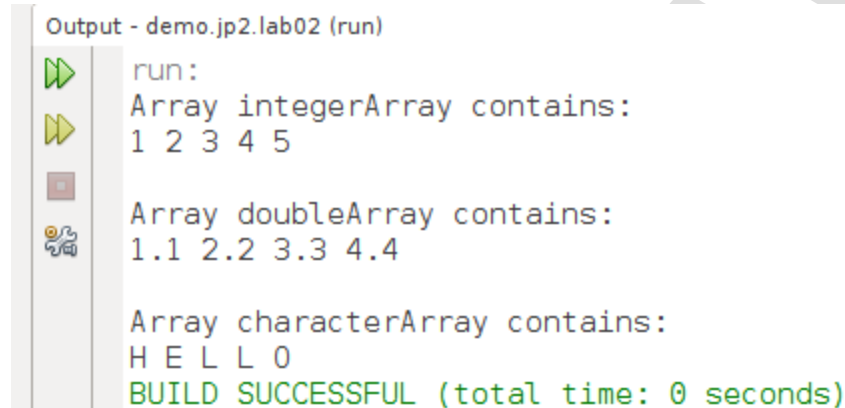
/**
 *
 * @author minhvuvc
 */
public class MainClass {
    // generic method printArray

    public static < E> void printArray(E[] inputArray) {
        // Display array elements
        for (E element : inputArray) {
            System.out.printf("%s ", element);
        }
        System.out.println();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
```

```
//Bước 2: Khởi tạo các mảng với kiểu dữ liệu khác nhau.  
Integer[] intArray = {1, 2, 3, 4, 5};  
Double[] doubleArray = {1.1, 2.2, 3.3, 4.4};  
Character[] charArray = {'H', 'E', 'L', 'L', 'O'};  
  
//Bước 4: Gọi hàm printArray() và truyền dữ liệu.  
System.out.println("Array integerArray contains:");  
printArray(intArray); // pass an Integer array  
  
System.out.println("\nArray doubleArray contains:");  
printArray(doubleArray); // pass a Double array  
  
System.out.println("\nArray characterArray contains:");  
printArray(charArray); // pass a Character array  
}  
}
```

Bước 5: Xem kết quả.



```
Output - demo.jp2.lab02 (run)  
run:  
Array integerArray contains:  
1 2 3 4 5  
  
Array doubleArray contains:  
1.1 2.2 3.3 4.4  
  
Array characterArray contains:  
H E L L O  
BUILD SUCCESSFUL (total time: 0 seconds)
```

**Bài thực hành 3:** Viết class AdvancedComparion là một Generics sử dụng kiểu dữ liệu T, trong class này viết hàm maximum nhận vào 3 tham số kiểu T và so sánh tìm ra giá trị lớn nhất. Viết lớp MainClass có hàm main(), khởi tạo đối tượng AdvanceCompare và truyền các giá trị kiểu int, float và String. Chạy chương trình và xem kết quả.

Bước 1: Viết class AdvancedComparion và hàm maximum().

```
package demo.jp2.lab02.phan1.bai3;  
  
/**  
 *  
 * @authorminhvuvc  
 */
```

```
public class AdvancedComparion<T extends Comparable<T>> {

    public void maximum(T a, T b, T c) {
        T max = a;
        if (b.compareTo(a) > 0) {
            max = b;
        }
        if (c.compareTo(b) > 0) {
            max = c;
        }
        System.out.println("Maximum is " + max);
    }
}
```

Bước 2: Tạo MainClass và trong hàm main() khởi tạo đối tượng AdvanceCompare, truyền các kiểu giá trị cho nó.

```
package demo.jp2.lab02.phan1.bai3;

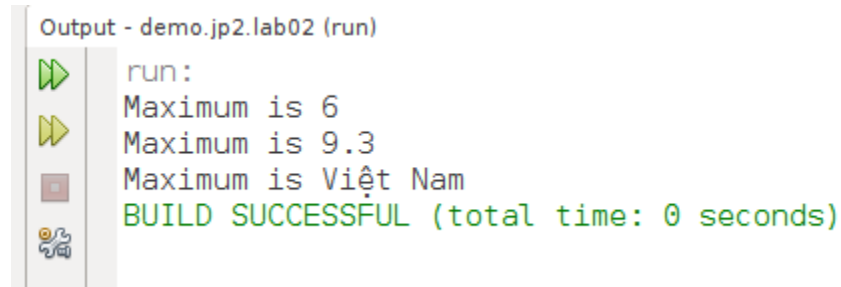
/**
 *
 * @authorminhvufc
 */
public class MainClass {

    /**
     * @param args thecommandlinearguments
     */
    public static void main(String[] args) {
        AdvancedComparison<Integer> compInt = new AdvancedComparison<>();
        AdvancedComparison<Float> compFloat = new AdvancedComparison<>();
        AdvancedComparison<String> compString = new AdvancedComparison<>();
    }
}
```



```
compInt.maximum(9, 3, 6);  
compFloat.maximum(8.5f, 9.3f, 2.6f);  
compString.maximum("Việt Nam", "Trung Quốc", "Hoa Kỳ");  
}  
}
```

Bước 3: Chạy chương trình.



Output - demo.jp2.lab02 (run)

```
run:  
Maximum is 6  
Maximum is 9.3  
Maximum is Việt Nam  
BUILD SUCCESSFUL (total time: 0 seconds)
```

## Phần IV - Bài tập tự làm

### Bài 1.

Tạo một collection dạng List để nhập vào một danh sách gồm n phần tử là số nguyên.  
Hiển thị danh sách vừa nhập ra màn hình  
Hiển thị danh sách sắp xếp tăng dần.

**Gợi ý:** sử dụng với ArrayList, cú pháp `List<Integer> lstSoNguyen = new ArrayList();`

### Bài 2.

Tạo một collection dạng Set để nhập vào tên của n thành phố  
Hiển thị danh sách vừa nhập  
Hiển thị danh sách sắp xếp tăng dần.

**Gợi ý:** Interface Set có thể khởi tạo dùng với các lớp như:

1. HashSet
2. TreeSet

Cú pháp ví dụ: `Set<String> words = new HashSet<String>();`

### Bài 3.

Tạo một collection dạng Map có key là kiểu Integer, Value là kiểu String.  
Nhập vào danh sách gồm n tên các quốc gia (key tăng dần từ 1 đến n)  
Hiển thị danh sách vừa nhập

Hiển thị danh sách sắp xếp tăng dần theo tên các quốc gia.

**Gợi ý:** Interface Map có thể khởi tạo dùng với các lớp như:

1. HashMap

2. TreeMap

Cú pháp ví dụ:

```
Map<String, EmployeeData> staffObj = new HashMap<String, EmployeeData>();  
staffObj.put("101", new EmployeeData("Anna John"));
```

#### Bài 4.

Tạo một lớp Book có các thuộc tính:

```
private String isbn;
```

```
private String bookName;
```

```
private String author;
```

```
private String publisher;
```

```
private float price;
```

- Tạo đầy đủ các constructor, các phương thức get/set.
- Cài đặt hàm toString() để hiển thị thông tin các thuộc tính
- Tạo lớp BookManager, trong lớp này khai báo một collection có kiểu Book.
- Nhập vào thông tin của n cuốn sách
- Hiển thị thông tin của n cuốn sách vừa nhập
- Hiển thị thông tin n cuốn sách sau khi sắp xếp giảm dần theo giá.

#### Bài 5.

Tạo lớp Product có các thuộc tính:

```
private String proId;
```

```
private String proName;
```

```
private String producer;
```

```
private int yearMaking;
```

```
private float price;
```

- Tạo đầy đủ constructor, các phương thức get/set, toString()
- Tạo lớp ProductTest để khai báo một collection dạng Map có key là kiểu Integer, value là kiểu Product
- Nhập vào n thông tin các product.

- Hiển thị thông tin vừa nhập.
- Hiển thị thông tin tăng dần theo năm sản xuất.

### **Bài 6.**

Cài đặt lớp Computer gồm các thuộc tính:

```
private String comId;
```

```
private String comName;
```

```
private String producer;
```

```
private float price;
```

- Cài đặt 2 constructor, các phương thức get/set.
- Cài đặt lớp GenericComputer<T>.
- Trong lớp này khai báo một collection List<T>
- Viết các hàm addComputer(T t) và displayComputer()
- Cài đặt lớp Test có hàm main, trong đó khai báo một đối tượng của lớp GenericComputer với kiểu T là Computer.
- Nhập vào thông tin cho n computers
- Hiển thị thông tin vừa nhập.

### **Bài 7.**

Cài đặt lớp GenericMoto để tạo generic cho collection kiểu Map.

```
public class GenericMoto<K,V>{
```

```
//...
```

```
}
```

- Trong lớp này khai báo một collection Map<K,V>.
- Constructor không có tham số để khởi tạo cho collection.
- Cài đặt hàm: public void addNewElement(K key, V value) để add thêm một phần tử cho collection.
- Cài đặt hàm: public void display() để hiển thị thông tin các phần tử của collection.
- Cài đặt lớp Test có hàm main. Trong lớp này khai báo một đối tượng của lớp GenericMoto, trong đó K có kiểu Integer, V có kiểu String.
- Nhập vào tên của n moto và hiển thị thông tin tên các moto vừa nhập.