ASSIGNMENT-12

1.Create a HashTable to maintain a bank detail which includes Account number and Customer name. Let Account number be the key in the HashTable. Write a Java program to implement the following operations in the HashTable

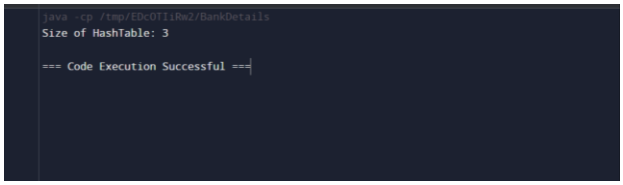
i) Add 3 records
ii) Display the size of HashTable
iii) Clear the HashTable

```
import java.util.Hashtable;
public class BankDetails {
   public static void main(String[] args) {
      Hashtable<Integer, String> bankDetails = new Hashtable<>();

      // Add 3 records
      bankDetails.put(123456, "Alice");
      bankDetails.put(789012, "Bob");
      bankDetails.put(345678, "Charlie");

      // Display the size of HashTable
      System.out.println("Size of HashTable: " + bankDetails.size());

      // Clear the HashTable
      bankDetails.clear();
   }
}
```



1. Create a employee record using map interface and do the following operations.

i. Add object iii. Remove specified object
ii. isEmpty or not iv. Clear

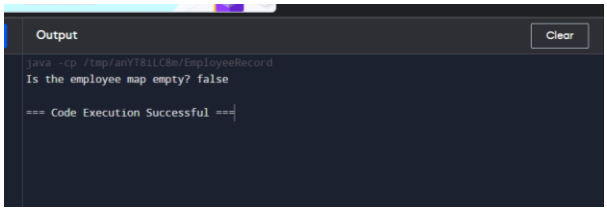
```
import java.util.HashMap;
import java.util.Map;

public class EmployeeRecord {
   public static void main(String[] args) {
      Map<Integer, String> employeeMap = new HashMap<>();
      employeeMap.put(1, "John Doe");
      employeeMap.put(2, "Jane Smith");
```

```
        System.out.println("Is the employee map empty? " + employeeMap.isEmpty());

        // Remove specified object
        employeeMap.remove(1);

        // Clear the map
        employeeMap.clear();
    }
}
```



2. Create a simple generics class with type parameters for sorting values of different types.

```
import java.util.Arrays;
public class GenericSort<T extends Comparable<T>> {
    private T[] array;
    public GenericSort(T[] array) {
        this.array = array;
    }
    public void sortArray() {
        Arrays.sort(array);
    }
   public void printArray() {
        for (T element : array) {
            System.out.print(element + " ");
        }
        System.out.println();
    }
    public static void main(String[] args) {
        Integer[] intArray = {4, 2, 7, 1, 5};
        GenericSort<Integer> intSort = new GenericSort<>(intArray);
        intSort.sortArray();
        intSort.printArray();

        String[] strArray = {"Java", "Python", "C", "JavaScript"};
        GenericSort<String> strSort = new GenericSort<>(strArray);
        strSort.sortArray();
        strSort.printArray();
    }
}
```

```
Output                                                    Clear
java -cp /tmp/HR3qarPtvo/GenericSort
1 2 4 5 7
C Java JavaScript Python

=== Code Execution Successful ===
```

3. Develop a Java code to insert the following elements, using ListIterator to append + symbol in each element and print them in reverse order. {C, A, E, B, D, F}.

```
import java.util.*;
public class ListIteratorExample {
    public static void main(String[] args) {
        List<String> elements = new ArrayList<>(Arrays.asList("C", "A", "E", "B", "D", "F"));
        ListIterator<String> iterator = elements.listIterator();
        while (iterator.hasNext()) {
            String element = iterator.next();
            iterator.set(element + "+");
        }
        while (iterator.hasPrevious()) {
            System.out.print(iterator.previous() + " ");
        }
    }
}
```

```
Output                                                    Clear
java -cp /tmp/AJHjdjNmbl/ListIteratorExample
F+ D+ B+ E+ A+ C+
=== Code Execution Successful ===
```

4. Generate a Java code to perform simple arithmetic operations and to throw Arithmetic Exception for Division-by-Zero.

```
public class ArithmeticOperations {
    public static void main(String[] args) {
        int num1 = 10;
        int num2 = 0;
        try {
            int result = num1 / num2;
            System.out.println("Division Result: " + result);
        } catch (ArithmeticException e) {
            System.out.println("Division-by-Zero Exception: " + e.getMessage());
        }
    }
}
```

Output

Clear

```
java -cp /tmp/W73ttBJV1o/ArithmeticOperations
Division-by-Zero Exception: / by zero

=== Code Execution Successful ===
```