The aim of this assignment is to gain proficiency in utilizing and incorporating generic types within Java classes.

Exercise 1 (acquire the expertise) – Tuples

Open the S1Es1.java file in your IDE. This program implements the Tuple class, which can store two Object instances and enables switching them by returning a new Tuple class with the contained objects swapped.

Using the provided source code as a starting point, follow these steps:

- Implement the Tuple1 class as a generic type, which provides a formal type parameter T for the stored objects. The swap method should return a new Tuple1 instance. Create example test cases for the Tuple1 class within the testTuple1Class method.
- Implement the Tuple2 class as a generic type, which defines two formal type parameters T1 and T2, allowing you to specify the types of the first and second object individually. The swap method in this class should also return a new Tuple2 instance. Implement example test cases for the Tuple2 class within the testTuple2Class method.

Exercise 2 (problem solving) – MyQueue

Write a program that implements a generic queue as a singly linked list of nodes. Each node should store a single value and link to the next node in the queue. The functionality of the queue is limited to the following operations:

- Add: creates a new node with the given value and appends it to the end of the queue.
- Remove: removes the head node of the queue and returns the value stored in that node.
- Print: prints all node values (using the toString() method) starting from the head of the queue.

To test your implementation, create a minimum of two queues of different types, and perform the following operations for each queue:

1. Add elements.
2. Print the queue.
3. Remove some elements.
4. Print the updated queue.

Exercise 3 (optional deepening) – Matrix

Develop a program that implements a generic Matrix class, which internally employs a two-dimensional array to store the matrix elements. The constructor for the Matrix class should accept the number of rows and columns as parameters to initialize the internal array. Make sure your program provides the following methods:

- set(row, col, value): this method stores the specified value at the provided row and column.
- get(row, col): it retrieves and returns the value stored at the specified row and column.
- print(): this function prints the entire matrix's contents, utilizing the toString() method of the object.
- transpose(): it returns a new instance of the Matrix class representing the transpose of the current matrix.

Verify the correctness of your implementation by creating a minimum of two Matrix instances, each containing different data types (such as Integer and String), and thoroughly test all available methods.