Generics Methods, Class and Collection

1. Write a generic method findMaxInArray that can accept an array of any type (e.g., Integer[], String[]) and find the maximum element. Avoid method overloading.

2. Create a generic class Pair that can hold two objects of potentially different types (e.g., Pair<String, Integer>). Implement getter methods for both elements. Create instances of Pair with different types and demonstrate their usage.

3. Create a generic LinkedList of integers. Add some numbers, then remove all even numbers from it

4. Create a generic ArrayList of Student objects (where Student has name and age). Sort this list by age using a custom Comparator

5. Write a generic method countUnique that takes a List of any type and returns the number of unique elements within that list. Utilize a HashSet internally to achieve this.

6. Use a generic HashMap to store product names and their quantities. Iterate through the map and print only those products with a quantity greater than 10.

7. Create a PriorityQueue for Integer objects and add several numbers. Then, extract them in ascending order to show how PriorityQueue maintains order.

8. Create a generic class MyStack that implements a basic stack data structure using an internal ArrayList. Implement push, pop, peek, and isEmpty methods.

9. Create a Student class with id, name, and grade properties. Use an ArrayList<Student> to store student records. Implement methods to add a student, remove a student by ID, find a student by ID, and list all students.

10. Create a Person class with attributes like name and age. Create an ArrayList of Person objects. Add several Person objects to the list. Sort the ArrayList of Person objects by age (you might need to implement Comparable or provide a Comparator). Filter the list to find all persons above a certain age.

11. Use a HashMap<String, Integer> to represent an inventory where keys are product names and values are quantities. Write functions to add new items, update quantities, and display the current inventory.