Generics and Comparisons - homework

1. List several advantages of using generics. If needed demonstrate with code

Type safety, less code and is more easily reused. Better performance.

1. List disadvantages of using generics. If needed demonstrate with code

Code using generics can become harder to read and understand, especially for beginners.

Complex generic hierarchies and wildcards (? extends T, ? super T) can make code less intuitive.

Type erasure.

Generics only work with reference types (e.g., Integer, String), not primitives (int, double, char).

6. Interoperability Issues

1. In MyArrayList class some functions have arguments (T element) and some functions have arguments(Object target) or (Object target, Object element). Why is this? What happens when we replace T argument with object argument and vice versa?

T ensure type safety, meaning only type t can be added or manipulated

If we declare the list as MyArrayList<String>, then add(T element) ensures that only String values can be added, avoiding accidental insertion of incompatible types.

 Methods like contains(Object target) or remove(Object target) accept Object because they **compare elements rather than insert them**.

 The target is simply an item we need to check against the stored elements, and Java allows us to compare objects of different types.

1. Implement the following class diagram and demonstrate use of generics.

+----------------+

| <<interface>> |

| Actor |

+----------------+

| + perform() |

| + speak() |

+----------------+

▲

│

--------------------------------

| |

| |

+----------------+ +----------------+

| Human | | Animal |

+----------------+ +----------------+

| - name: String | | - species: String |

| - age: int | | - sound: String |

| - profession: String | + perform() |

| + perform() | | + speak() |

| + speak() | | + eat() |

| + work() | +----------------+

+----------------+

+--------------------------------+

| Circus<T extends Actor> |

+--------------------------------+

| - performers: List<T> |

| + addPerformer(T) |

| + showPerformance() |

+--------------------------------+

1. Implement the following classes and make objects Plane and Passenger Comparable and also implement Comparator in another class where you can compare passengers and planes also check if a passenger has a ticket reserved on a certain plane. Also sort passengers and planes using different attributes. Demonstrate interface comparable and comparator using different attributes.

Tip: use Collections.sort method here.

+------------------+

| Plane |

+------------------+

| - airline: String |

| - totalSeats: int |

| - destination: String |

| - from: String |

| - flightNo: String |

+------------------+

|

| 1

|---------------------

| 0..\* (has)

v

+------------------+

| Passenger |

+------------------+

| - name: String |

| - age: int |

| - numOfPeople: int |

| - typeOfSeat: SeatType |

+------------------+

|

| (uses)

v

+------------------+

| SeatType (enum) |

+------------------+

| + Premium |

| + Economy |

| + Business |

+------------------+

Lecture 25 feb – Arraylist and Linkedlist

Lecture 27 feb – Doubly linkedlist