National University of Computer and Emerging Sciences

Software Construction and Development

Lab Manual 2

Muhammad Hassan Raza

Fall 2024

Department of Software Engineering
FAST-NU, Lahore, Pakistan

Problem 1: Create a generic management system that can manage different types of tasks or jobs using a simple FIFO queue and allows sorting using custom criteria.

1) Create a generic class `SimpleQueue<T>` with the following properties:
   a) `queue` (ArrayList of `T`)

2) The class should have the following methods:
   a) `void enqueue(T element)`: Adds an element to the end of the queue.
   b) `T dequeue()`: Removes and returns the element from the front of the queue.
   c) `List<T> getSortedList(Comparator<T> comparator)`: Returns a sorted list of elements based on a custom comparator.
   d) `void displayQueue()`: Prints all elements in the queue.

3) Create a class `Task` with the following properties:
   a) `name` (String)
   b) `description` (String)

4) Create another class `Job` with the following properties:
   a) `title` (String)
   b) `complexityLevel` (int)

5) Create a `main` method where you:
   a) Instantiate two `SimpleQueue` objects: `SimpleQueue<Task>` and `SimpleQueue<Job>`.
   b) Add several `Task` objects with different names and descriptions, and `Job` objects with different titles and complexity levels to their respective queues.
   c) Use the `dequeue()` method to remove and display the first element from each queue.
   d) Use `getSortedList()` with custom comparators that sort `Task` objects by `name` and `Job` objects by `title`.
   e) Display the sorted lists.

Example Output:

--- Task Queue ---
Task added: Task A (Description: "Do homework")
Task added: Task B (Description: "Clean the house")
Task added: Task C (Description: "Write a report")

Original Task Queue:
Task A (Description: "Do homework")
Task B (Description: "Clean the house")
Task C (Description: "Write a report")

Dequeued Task: Task A (Description: "Do homework")

Queue after dequeuing the first task:
Task B (Description: "Clean the house")
Task C (Description: "Write a report")

Sorted Tasks by Name:
Task A (Description: "Do homework")
Task B (Description: "Clean the house")
Task C (Description: "Write a report")

--- Job Queue ---
Job added: Job X (Complexity Level: 5)
Job added: Job Y (Complexity Level: 2)
Job added: Job Z (Complexity Level: 4)

Original Job Queue:
Job X (Complexity Level: 5)
Job Y (Complexity Level: 2)
Job Z (Complexity Level: 4)

Dequeued Job: Job X (Complexity Level: 5)

Queue after dequeuing the first job:
Job Y (Complexity Level: 2)
Job Z (Complexity Level: 4)

Sorted Jobs by Title:

Job X (Complexity Level: 5)
Job Y (Complexity Level: 2)
Job Z (Complexity Level: 4)

Problem 2: Create a generic shape management system that calculates the area of different shapes using inheritance and generics.

1) Create a generic abstract class `Shape<T extends Number>` with the following properties:
   a) `dimensions` (ArrayList of `T`)

2) The `Shape` class should have the following abstract methods:
   a) `double calculateArea()`: Calculates and returns the area of the shape.
   b) `void display()`: Displays the shape type and its dimensions.

3) Create three subclasses `Rectangle`, `Circle`, and `Triangle` that extend `Shape`:
   a) `Rectangle<T extends Number>`: Represents a rectangle with dimensions `length` and `width`.
   b) `Circle<T extends Number>`: Represents a circle with `radius`.
   c) `Triangle<T extends Number>`: Represents a triangle with base `base` and height `height`.

   d) All three classes should implement the `calculateArea()` and `display()` methods.

4) Create a generic utility class `ShapeUtils` with the following static generic method:
   a) `public static <T extends Number> void printArea(Shape<T> shape)`: Prints the area of a given shape.

5) In the `main` method:
   a) Instantiate `Rectangle<Double>`, `Circle<Integer>`, and `Triangle<Float>` objects with appropriate dimensions.
   b) Calculate and display the area for each shape using `printArea()`.

Expected Output Example:

Rectangle with Length: 5.5 and Width: 3.2
Area of Rectangle: 17.6

Circle with Radius: 7
Area of Circle: 153.9

Triangle with Base: 5.0 and Height: 8.0
Area of Triangle: 20.0