Practice 3

Deadline: 2 weeks from now. Should be checked onsite (during labs).

Task 1

Implement a task scheduling system with `Task` and `TaskScheduler`. Each task has a priority and a description. The `TaskScheduler` should:

- Manage tasks using a `PriorityQueue` based on priority (higher priorities first). If two tasks have the same priority, order them lexicographically by description.
- Implement the following methods:
    1. `void addTask(String description, int priority)`: Adds a new task to the task scheduler.
    2. `List<Task> getTopKTasks(int k)`: Retrieves the top K tasks with the highest priorities *without modifying the queue*.
    3. `void finishNextTask()`: Finishes (removes) the task with the highest priority.

Sample test code:

```
public class TaskSchedulerTest {
    public static void main(String[] args) {
        TaskScheduler scheduler = new TaskScheduler();
        scheduler.addTask("Write report", 2);
        scheduler.addTask("Respond to emails", 1);
        scheduler.addTask("Prepare presentation", 3);
        scheduler.addTask("Code review", 2);
        scheduler.addTask("Team meeting", 5);
        scheduler.addTask("Project planning", 4);
        scheduler.addTask("Client follow-up", 3);
        scheduler.addTask("Bug fixing", 2);
        scheduler.addTask("Lunch break", 1);
        scheduler.addTask("Team outing", 1);

        System.out.println("Top 5 priority tasks:");
        List<Task> top5Tasks = scheduler.getTopKTasks(5);
        top5Tasks.forEach(e -> System.out.println(e));

        System.out.println("\nFinishing the next 3 highest priority tasks\n");
        scheduler.finishNextTask();
        scheduler.finishNextTask();
        scheduler.finishNextTask();

        System.out.println("Top 6 priority tasks:");
        List<Task> top6Tasks = scheduler.getTopKTasks(6);
        top6Tasks.forEach(e -> System.out.println(e));
    }
}
```

Sample output:

```
Top 5 priority tasks:
Task{description='Team meeting', priority=5}
Task{description='Project planning', priority=4}
Task{description='Client follow-up', priority=3}
Task{description='Prepare presentation', priority=3}
Task{description='Bug fixing', priority=2}

Finishing the next 3 highest priority tasks

Top 6 priority tasks:
Task{description='Prepare presentation', priority=3}
Task{description='Bug fixing', priority=2}
Task{description='Code review', priority=2}
Task{description='Write report', priority=2}
Task{description='Lunch break', priority=1}
Task{description='Respond to emails', priority=1}
```

Unlike our tutorial that `sorts` items of a `List`, you should leverage the APIs of the PriorityQueue for this task.

Task 2

In this task, you'll write a program that allows users to filter any integer list based on a certain criterion. Specifically, the users may choose one of the functions as the predicate for filtering:

- Get only even numbers
- Get only odd numbers
- Get only prime numbers

For this purpose, we may define a functional interface, `MyPredicate`, which is used for testing whether a given object `t` meets a certain criterion.

```
public interface MyPredicate<T> {
    boolean test(T t);
}
```

Then, we can define a **higher-order function**, `filter`, which takes a list and a predicate (i.e., a function for filtering), and returns the filtered list.

```
public <T> List<T> filter(List<T> list, MyPredicate<T> p) {
    // TODO
}
```

Please complete the `filter` function, and use it to write a program that allows users to choose a function, input an integer list, and print the results after applying the function.

Sample Output

```
Please input the function no:
1 - Get even numbers
2 - Get odd numbers
3 - Get prime numbers
0 - Quit
2
Input the integer list:
1 2 3 4 5 6
Filter results:
[1, 3, 5]
Please input the function no:
1 - Get even numbers
2 - Get odd numbers
3 - Get prime numbers
0 - Quit
1
Input the integer list:
2 3 4 5
Filter results:
[2, 4]
Please input the function no:
1 - Get even numbers
2 - Get odd numbers
3 - Get prime numbers
0 - Quit
3
Input the integer list:
1 2 3 4 5 6 7
Filter results:
[2, 3, 5, 7]
Please input the function no:
1 - Get even numbers
2 - Get odd numbers
3 - Get prime numbers
0 - Quit
0
```

Evaluation

The practice will be checked by teachers or SAs. What will be tested:

1. That you understand every line of your own code, not just copy from somewhere
2. That your program compiles correctly (javac)
3. Correctness of the program logic
4. That the result is obtained in a reasonable time

Late submissions after the deadline will incur a 20% penalty, meaning that you can only get 80% of this practice's score.