Implement a task executor service according to the following specification.

The entry point for the service is Task Executor interface. The interface is defined bellow including its dependencies.

The service is required to implement the following behaviors:

1. Tasks can be submitted concurrently. Task submission should not block the submitter.
2. Tasks are executed asynchronously and concurrently. Maximum allowed concurrency may be restricted.
3. Once task is finished, its results can be retrieved from the Future received during task submission.
4. The order of tasks must be preserved.
   * The first task submitted must be the first task started.
   * The task result should be available as soon as possible after the task completes.
5. Tasks sharing the same TaskGroup must not run concurrently.

Additional implementation requirements:

1. The implementation must run on OpenJDK 17.
2. No third-party libraries can be used.
3. The provided interfaces and classes must not be modified.

Please, write down any assumptions you made.

import java.util.UUID;

import java.util.concurrent.Callable;

import java.util.concurrent.ExecutionException;

import java.util.concurrent.Executors;

import java.util.concurrent.Future;

public class Main {

  /\*\*

   \* Enumeration of task types.

   \*/

  public enum TaskType {

    READ,

    WRITE,

  }

  public interface TaskExecutor {

    /\*\*

     \* Submit new task to be queued and executed.

     \*

     \* @param task Task to be executed by the executor. Must not be null.

     \* @return Future for the task asynchronous computation result.

     \*/

    <T> Future<T> submitTask(Task<T> task);

  }

  /\*\*

   \* Representation of computation to be performed by the {@link TaskExecutor}.

   \*

   \* @param taskUUID Unique task identifier.

   \* @param taskGroup Task group.

   \* @param taskType Task type.

   \* @param taskAction Callable representing task computation and returning the result.

   \* @param <T> Task computation result value type.

   \*/

  public record Task<T>(

    UUID taskUUID,

    TaskGroup taskGroup,

    TaskType taskType,

    Callable<T> taskAction

  ) {

    public Task {

      if (taskUUID == null || taskGroup == null || taskType == null || taskAction == null) {

        throw new IllegalArgumentException("All parameters must not be null");

      }

    }

  }

  /\*\*

   \* Task group.

   \*

   \* @param groupUUID Unique group identifier.

   \*/

  public record TaskGroup(

    UUID groupUUID

  ) {

    public TaskGroup {

      if (groupUUID == null) {

        throw new IllegalArgumentException("All parameters must not be null");

      }

    }

  }

}