Trail Running Competition Management

Develop the core of an application to manage a trail running competition The program classes are in the `trail` package and a sample main is in the `main` package. The façade class of the program is `Trail`.

You can access a copy of the JDK documentation on a local server.

R1. Runners

The participants can register to the competition through the method `newRunner()` that accepts name and surname, and returns the bib number. The numbers are assigner progressively starting from 1.

It is possible to access a runner's information by means of `getRunner()` that is provided in two distinct versions; the first one accepts the bib number and returns a `Runner` object; the second one accepts a surname and returns a collection of runners (all those sharing the given surname, sorted by ascending bib number).
The full list of participants, sorted by bib number, is available through method `getRunners()`. While the list sorted alphabetically is available through method `getRunnersByName()`, in case of homonyms the runner with the smalles bid number comes first.

The class *`Runner`* allows accessing the information about the runner - bib number, name, surname - through the relative getter methods.

R2. Path

The competition path is defined as a series of check location.
The locations are defined using method `addLocation()` that accepts as argument the unique name of the location.
Locations must be added in order, starting from the first -- corresponding to the departure -- up to the latest -- corresponding to the arrival --.

It is possible to access the information about a location using method `getLocation()` that accepts a name and returns a `Location` object. The full path can be accessed through method `getPath()` that returns the sorted list of locations.
Class *`Location`* provides the getter methods to retrieve the name and the order number of the location (starting at 0).

R3. Delegates

The trail path is checked by several competition organizing committee delegates.
The delegates are registered using `newDelegate()` that receives as arguments name, surname, and SSN id of the person.
The `getDelegates` returns the list of delegates (in the form *`"Surname, Name, SSN"`*) sorted alphabetically.
The delegates can be assigned to locations along the path using `assignDelegate()` that accepts the name of the location and the SSN id. It is possible for a location to have several assigned delegates, and for a delegate to be assigned to different locations. In case of wrong SSN id or position name the method throws a `TrailException`.

Given a location, it is possible to access the assigned delegates using `getDelegates()` that accepts the name of the location and returns a list of delegates (with the above format) sorted alphabetically.

R4. Controls

The delegates' task is to record the passage of a runner at a specific location.
The recording is performed using `recordPassage()` that accepts as arguments the SSN id of the delegate, the name of the location, and the bib number of the runner.
The method must check the existence of delegate, location, and bib number; in case of error it throws a *`TrailException`*.
In case of success the system stores as time of passage the current system time (retrieved using *`System.currentTimeMillis()`*) and returns the same time.

For each runner, the passage time can be obtained using `getPassTime()` that accepts the name of the position and the bib number, and returns the time of passage, or -1 in case no passage had been recorded yet by the runner at the location.
It the name of the position or the bib number are not correct a *`TrailException`* is thrown.

R5. Statistics

The method `getRanking()` can be used to retrieve the order of passage at a specific location, it accepts the name of the location and returns a list of runners, sorted by ascending passage time, first the runner who passed first at the location.

The full ranking can be retrieved using the `getRanking()` with no arguments; the ranking is built taking into account the most advanced locations the runners have reached and the relative passage times.

For instance if the trail path is consists of three locations (Courmayeur, La Thuile, and Valgrisanche) in that order, and the passage times are those reported in the table, the raking would be 2, 3, 1, 4.

Bib	Latest location	Time
1	La Thuile	14:20
2	Valgrisanche	19:10
3	La Thuile	14:10
4	Courmayeur	10:05