Mars Rover

The program is built in Java 8, making use of the Object-Oriented Programming techniques.

Assumptions and deductions:

- A rover cannot overlap another rover on the plateau. The location of a rover changes only if the resulting location does not overlap another rover, otherwise remains unchanged.
- Once present on the plateau, a rover is not removed, but remains in the grid while other rovers are added and moved
- A rover's location has to respect the grid limits. The location changes as desired only if the resulting location respects the grid limit, otherwise remains unchanged.
- A rover's real size could impose more other conditions. Here we assume that a rover's location is enough to infer its space occupation on the grid.
- From the information provided, a rover does not move backwards.
- When turning to left or right, there is always 90 degrees change.

Implementation

- **Rover Class** - Every rover is an instance of the class Rover, its main characteristics being its name, the coordinates forming the location, and the orientation.
  - A rover can change its orientation
  - A rover can move one point on the grid, provided that its future location will respect the grid limits and will not overlap another existing rover. Otherwise, the location is not changed.
  - A rover can follow an instruction (move a point on the grid, change orientation), depending on the specified command (L/R/M), taking into consideration the requirement imposed by each.
  - A rover can follow an instruction set, for each command performing the necessary checks before action.
  - A rover's location can be changed by providing a new location.
- **The Location Class**
  - A location holds the (x, y) coordinates
  - The x and y coordinates can be changed by providing a value to add (if negative, the coordinates will decrease in value). This is used in the Rover class when moving on point on the grid.
- **The Direction Class**
  - A direction holds its name, the left and right adjacent directions.
  - When turning left or right, a new direction instance is created and returned, holding the new name, left and right adjacent directions. This is called from the Rover Class when a rover is changing its orientation.
- **The Main Class**
  - The program input is read from a text file present in the project's directory.
  - A map of locations continuously being kept track of, every new rover's location is added and updated following instructions.
  - The grid upper right coordinates are extracted, and the grid limits are set.
  - For each rover input in the file, its coordinates and orientation are extracted and used to create a new Rover instance, its location being added to the map.

    - The list of instruction is extracted and followed; the specific checks being made at every instruction point.
    - Once the instruction set is followed the resulting position and orientation is printed out.
    - The process is repeated for each rover input and finishes once there is no input in the file.

Run Instructions

- The unzipped folder can be run in an integrated environment such us IntelliJ IDEA.
- The input is extracted from the input.txt file in the project directory. This path is always automatically updated, regardless of the project's location on the computer the program is run on.
- The results are printed out to the console.