**To Run Rover:**

**Compile: execute *ant* in {MarsRover} directory**

**Run: execute *ant run* in {MarsRover} directory.**

**Interfaces and Classes:**

1. **Instruction (Interface):** This interface is implemented by **InstructionL**, **InstructionR** and **InstructionM** classes. Each of these classes contains logic to execute instructions **L**, **R** and **M** respectively**.** These classes implements “executeInstruction” method which takes **Rover** object

and executes instruction over it.

1. **Rover (Class):** This class encapsulates rover object. It defines rover coordinates (x,y coordinates and head direction). It also encloses plateau dimensions and rover move strategy which defines how a rover moves when given instructions.
2. **Instruction Factory (Interface):** This interface defines contract for implementing instruction factory which creates objects of **Instruction** type. This interface is implemented by class **DefaultInstructionFactory.**
3. **Constants (Class): This class represents all four directions with their angle with +X axis.**
4. **TestRover: This class implements the main method and is used to test this program.**

**How Rover Moves:**

**For moving a rover its “moveRover” method is called with the instruction string (LMLMLMLMMM) . “moveRover” method in turn calls “moveRover” method of DefaultMovingStrategy which converts instruction string commands into corresponding objects of instruction implementations (objects of InstructionL**, **InstructionR** and **InstructionM** classes**). Now “execteInstruction” method of each instruction object is called which takes care of moving the rover.**

**Rover’s Coordinate Calculations:**

**North –South directions are taken along +Y and –Y axis. Similarly West-East directions are taken along +X and –X axis.**

**North direction makes 90 degrees with +X –axis.**

**West direction makes 180 degrees angle with +X axis.**

**South direction makes 270 degrees angle with +X axis.**

**East direction makes 0 degrees angle with +X axis.**

**In this program rover head direction is defined by degrees. Suppose rover wants to move one step in its head direction then its new coordinates are calculated as:**

**xNew=xOld+1\*cos(head direction) and yNew=yOld+1\*sin(head direction).**

**Rover’s Direction Calculations:**

**For executing instruction “L”, rover’s head direction is incremented by 90 degrees as:**

**headDirection =(headDirection+90)%360.**

**(This logic is present in “InstructionL” class)**

**For executing instruction “R”, rover’s head direction is decremented by 90 degrees as:**

**headDirection =headDirection-90**

**IF headDirection<0 then headDirection= headDirection+360;**