Enums in Java → Robot control

Medium 19 minutes

156 users solved this problem. Latest completion was about 19 hours ago.

Wow! This problem is kind of tricky. If you're ready to put your thinking cap on, brace yourself and good luck! Otherwise, you can skip it for now and return any time later

There is a robot in the game field. The position of the robot in this field is described by two integer coordinates: X and Y. The *X* axis is oriented from left to right, the *Y* axis — from bottom to top.

At the initial moment, the robot is in some coordinate on the field. It's also known where the robot looks: up, down, to the right or to the left. The initial position of the robot and its direction can have **any values**. You need to bring the robot to the destination point of the game field.

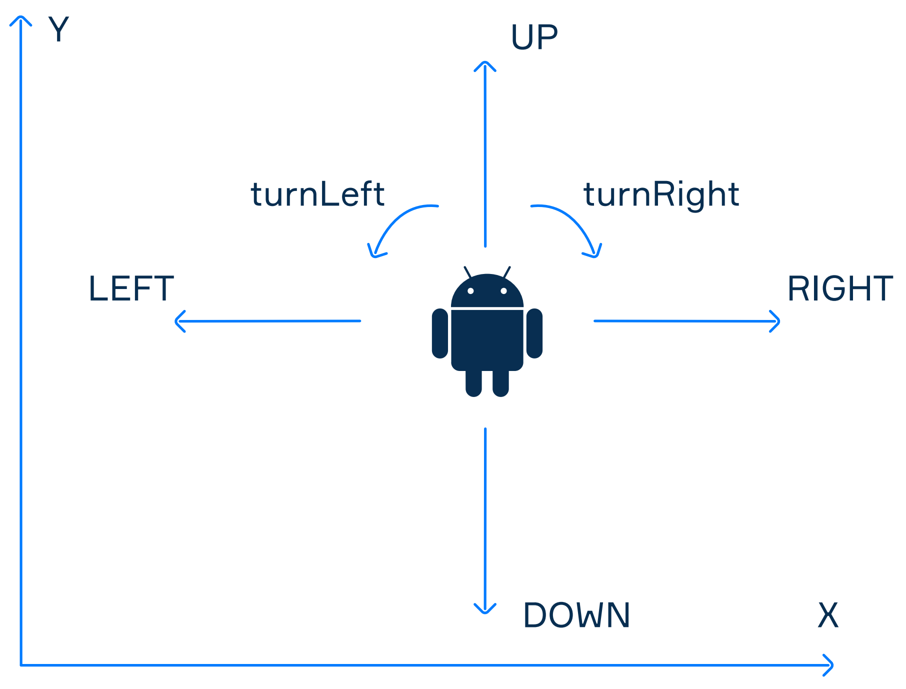
A robot is described by the `Robot` class. You can use the following methods of this class (with unknown implementation):

```
public class Robot {

    public Direction getDirection() {
        // current direction
    }

    public int getX() {
        // current X coordinate
    }



    public int getY() {

        // current Y o

    }



    public void turnLeft() {

        // rotate the robot 90 degrees counterclockwise

    }



    public void turnRight() {

        // rotate the robot 90 degrees clockwise

    }



    public void stepForward() {

        // take one step in the current direction

        // x or y coordinate will be changed by 1

    }

}
```

The direction of the robot is an enumeration:

```
public enum Direction {
    UP,
    DOWN,
    LEFT,
    RIGHT
}
```

It looks like the picture below:



Example

- The following values are passed to the method: toX == 3, toY == 0.
- The initial state of this robot: robot.getX() == 0, robot.getY() == 0, robot.getDirection() == Direction.UP.

To bring the robot to the destination point (3, 0), the method should call the following methods:

```
robot.turnRight();
robot.stepForward();
robot.stepForward();
robot.stepForward();
```

Another Example

- The following target values are passed to the method: toX == 0, toY == -1.
- The initial state of this robot: robot.getX() == 1, robot.getY() == 1, robot.getDirection() == Direction.RIGHT.

To bring the robot to the destination point (0, -1), the method should call the following methods:

```
robot.turnRight();
robot.turnRight();
robot.stepForward();
robot.turnLeft();
robot.stepForward();
robot.stepForward();
```

Try to crack this problem!

Hint

Report a typo

You've seen the solution so this problem will not be added to your progress. Solve an additional problem to complete the topic.

Write a program

Code Editor IDE

Java

```
class Move {
    public static void moveRobot(Robot robot, int toX, int toY) {
        robot.stepForward(); // your implementation here
    }
}

//Don't change code below

enum Direction {
    UP(0, 1),
    DOWN(0, -1),
    LEFT(-1, 0),
    RIGHT(1, 0);
```

```java

    private final int dx;
    private final int dy;

    Direction(int dx, int dy) {
        this.dx = dx;
        this.dy = dy;
    }

    public Direction turnLeft() {
        switch (this) {
            case UP:
                return LEFT;
            case DOWN:
                return RIGHT;
            case LEFT:
                return DOWN;
            case RIGHT:
                return UP;
            default:
                throw new IllegalStateException();
        }
    }

    public Direction turnRight() {
        switch (this) {
            case UP:
                return RIGHT;
            case DOWN:
                return LEFT;
            case LEFT:
                return UP;
            case RIGHT:
                return DOWN;
            default:
                throw new IllegalStateException();
        }
    }

    public int dx() {
        return dx;
    }

    public int dy() {
        return dy;
    }
}

class Robot {
    private int x;
    private int y;
    private Direction direction;

    public Robot(int x, int y, Direction direction) {
        this.x = x;
        this.y = y;
        this.direction = direction;
    }

    public void turnLeft() {
        direction = direction.turnLeft();
    }

    public void turnRight() {
        direction = direction.turnRight();
    }

    public void stepForward() {
        x += direction.dx();
        y += direction.dy();
    }

    public Direction getDirection() {
        return direction;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }
}
```

Run Continue Solutions (26)

Time limit: 8 seconds Memory limit: 256 MB

Comments (5) Hints (1) Useful links (0) Solutions (26) Show discussion