Karel J. Robot Chapter 2

move()

When a robot is sent a move message it executes a move instruction and moves forward one block; it continues to face the same direction. To avoid damage, a robot will not move forward if it sees a wall section or boundary wall between its current location and the corner to which it would move. Instead, it performs an error shutoff.

turnLeft()

When a robot is sent a turnLeft message it executes a turnLeft instruction by pivoting 90° to the left. The robot remains on the same street corner while executing a turnLeft instruction. Because it is impossible for a wall section to block a robot's turn, turnLeft cannot cause an error shutoff. Karel-Werke's designer purposely did not provide a built-in turnRight instruction.

pickBeeper()

When a robot is sent a pickBeeper message, it executes a pickBeeper instruction. It picks up a beeper from the corner on which it is standing and then deposits the beeper in its beeper-bag. If a pickBeeper instruction is attempted on a beeperless corner, the robot performs an error shutoff. On a corner with more than one beeper the robot picks up one, and only one, of the beepers and then places it in the beeper-bag.

putBeeper()

When a robot is sent a putBeeper message, it executes a putBeeper instruction by extracting a beeper from its beeper-bag and placing the beeper on the current street corner. If a robot tries to execute a putBeeper instruction with an empty beeper-bag, the robot performs an error shutoff. If the robot has more than one beeper in its beeper-bag, it extracts one, and only one, beeper and places it on the current corner.

Beepers are so small that robots can move right by them; only wall sections and boundary walls can block a robot's movement. Robots are also very adept at avoiding each other if two of more show up on the same corner simultaneously.

So far, we have seen several instructions that can cause error shutoffs. Make sure the following conditions are always satisfied:

- A robot executes a **move** instruction only when the path is clear to the next corner immediately in front of it.
- A robot executes a **pickBeeper** instruction only when it is on the same corner as at least one beeper.
- A robot executes a **putBeeper** instruction only when the beeper-bag is not empty.

Robot Descriptions

The formal name for the description of a robot instruction is **method**. The simple model of robot described above is called the **UrRobot** (ur is a German prefix meaning "original" or "primitive." The pronunciation of ur is similar to the sound of "oor" in "poor." class.) The specifications of the **UrRobot** class in the robot programming language follows.

```
class UrRobot
{
        public void move()
        {...}
        public void turnLeft()
        {...}
        public void pickBeeper()
        {...}
        public void putBeeper()
        {...}
}
```

Following the class name is a list of methods for this kind of robot. The list is always written in braces **{** and **}**. The use of elipsis in the above {...} indicates that there are some things that belong here that are not being shown. These are the descriptions of how an UrRobot would carry out each of these instructions.

The word **void** prefixes each of these methods to indicate that they return no feedback when executed. Later we will see additional methods that do produce feedback when executed, rather than changing the state of the robot as these instructions all do. Said differently, **void** means that these instructions do something as opposed to telling us something. The matching parentheses that follow the method names mark them as the names of actions that a robot will be able to carry out.

Note that a class is not a robot. It is just a description of robots of the same kind. The class of a robot tells us its capabilities (methods). We will now create an actual robot.

A Complete Program

```
import kareltherobot.*;

public class Sample implements Directions
{
    public static void main(String [] args)
    {
        World.readWorld("Sample.kwld");
        World.setVisible(true);
        World.setDelay(8);
        UrRobot karel = new UrRobot(1, 2, East, 0);
        karel.move();
        karel.move();
        karel.pickBeeper();
        karel.move();
        karel.turnLeft();
        karel.move();
        karel.move();
        karel.putBeeper();
        karel.move();
    }
}
```

A Task for Two Robots

Karel should carry the beeper to Carl and put it down. Carl should then pick it up and carry it to 1st Street and 3rd Avenue. The beeper should be placed on this corner. Both robots should face East at the end.

```
        UrRobot karel = new UrRobot(3, 1, East, 0);
        UrRobot carl  = new UrRobot(1, 1, East, 0);
        karel.pickBeeper();
        karel.turnLeft();
        karel.turnLeft();
        karel.turnLeft();
        karel.move();
        karel.move();
        karel.turnLeft();
        karel.putBeeper();
        carl.pickBeeper();
        carl.move();
        carl.move();
        carl.putBeeper();
```

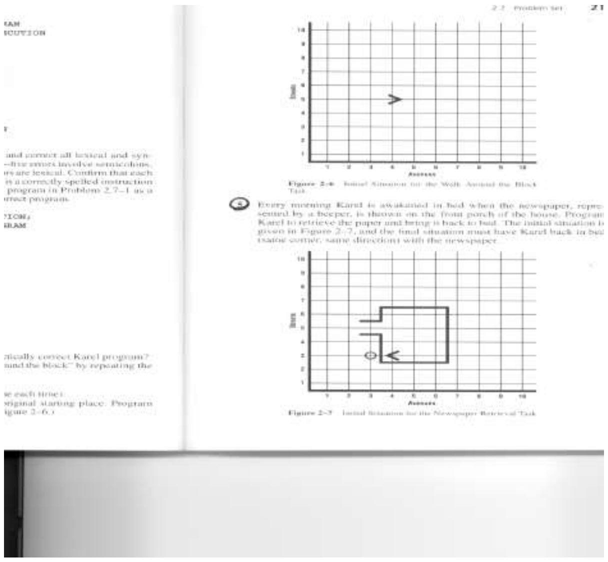
An infinity of Beepers

A robot can be delivered with infinitely many beepers in its beeper-bag. If such a robot puts down a beeper or picks a beeper, the number of beepers in the beeper-bag does not change. It is still infinity.

```
UrRobot Karel = new UrRobot(3, 2, East, infinity);
```

Problem Set

1. NewspaperRetrieval: Every morning karel is awakened in bed when the newspaper (represented by a beeper) is thrown on the front porch of the house. Program karel to retrieve the paper and bring it back to bed. The initial situation is shown in the figure below and the final situation must have karel back in bed (same corner, same direction) with the newspaper.



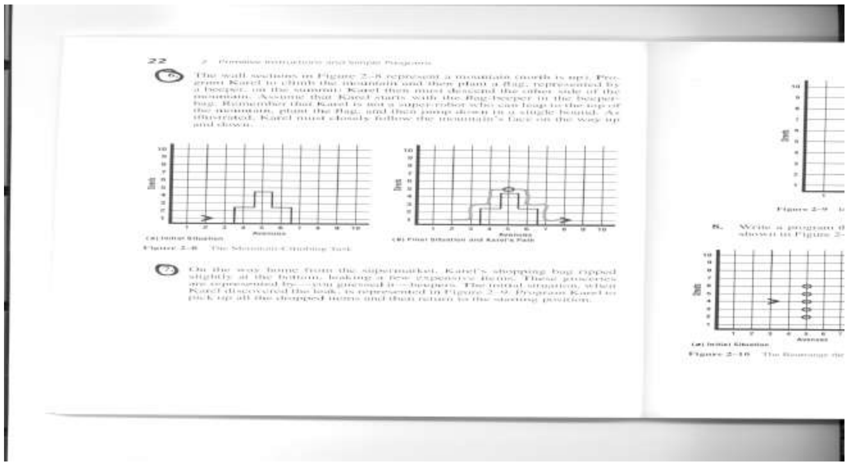
[illegible]

Figure 2-6 Initial Situation for the Walk Around the Block Task

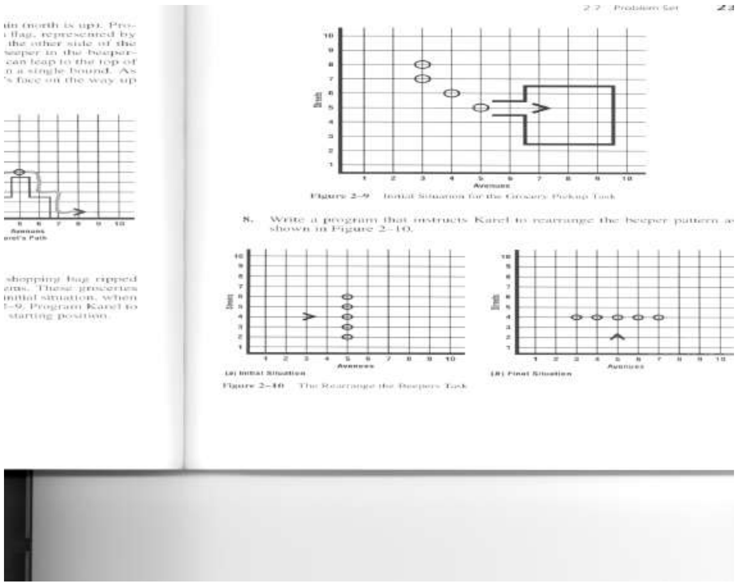
5. Every morning Karel is awakened in bed when the newspaper, represented by a beeper, is thrown on the front porch of the house. Program Karel to retrieve the paper and bring it back to bed. The initial situation is given in Figure 2-7, and the final situation must have Karel back in bed (same corner, same direction) with the newspaper.

Figure 2-7 Initial Situation for the Newspaper Retrieval Task

2. MountainClimbing: The wall sections in the figure below represent a mountain. Program joe to climb the mountain and plant a flag (a beeper) on the summit. Joe then must descend the other side of the mountain. Joe is not a super-robot who can leap to the top of the mountain, plant the flag, and then jump down in a single bound. He must closely follow the mountain's face.



3. GroceryPickup: On the way home from the supermarket, sara's shopping bag ripped slightly at the bottom, leaking a few expensive items. These groceries are represented by beepers. The initial situation when sara discovered the leak is shown below. Sara has to pick up all the dropped items and then return to the starting position.



4. PassBeepers: A robot named mike is at the origin (1,1) facing North with one beeper in its beeper bag. Three blocks East of mike, is another robot named amy (1,4) who is also facing North with no beepers. Have mike walk to amy and give amy the beeper. Amy should then carry the beeper two blocks North and put it down. Amy should walk back to where mike is waiting for her.