My Solution on Assignment 1 - Problem 4

This is my Solution of Assignment 1 - Problem 4 from the Computer Science Course CS106A of Prof. Mehran Sahami at the STANFORD University.

MidpointFindingKarel.java

```
/*
 * File: MidpointFindingKarel.java
 * -------------------------------
 * When you finish writing it, the MidpointFindingKarel class should
 * leave a beeper on the corner closest to the center of 1st Street
 * (or either of the two central corners if 1st Street has an even
 * number of corners).  Karel can put down additional beepers as it
 * looks for the midpoint, but must pick them up again before it
 * stops.  The world may be of any size, but you are allowed to
 * assume that it is at least as tall as it is wide.
 */

import stanford.karel.*;

public class MidpointFindingKarel extends SuperKarel {

    public void run() {
        /*
         * Karel finds the middle (-+1) of the world and drops
         * a beeper there.
         */
        checkWidth();
        goMiddle();
        putBeeper();
        paintCorner(RED);

        while( beepersPresent() ) { // infinite loop (dance)
            paintCorner(GREEN);
            turnLeft();
            paintCorner(RED);
            turnLeft();
        }
    }

    private void checkWidth() {
        /* this method lays a line of beepers to determine the width of
his world
         * then it stacks/stores the beepers at 1,1
         *
         * Pre-condition:    Karel at 1,1 facing EAST
         *                     no Beepers at all
         *
```

```
         * Post-condition:    Karel at 1,1 facing EAST
         *                     Width of World represented as a Stack of
         *                     Beepers at 1,1
         */

        drawLine();
        collectBeepers();
    }


    private void drawLine() {
        /* this method lays a line of beepers to determine the width of
his world
         *
         * Pre-condition:    Karel at 1:1 facing EAST
         *                     no Beepers at all
         * Post-condition:    Karel infront of RIGHT WALL (x,1) facing
EAST
         *                     behind him a line of beepers
         */
        while( frontIsClear() ) {
            putBeeper();
            move();
        }
        putBeeper();
    }

    private void collectBeepers() {
        /* this method collects a line of beepers and stacks them
         * at 1,1
         *
         * Pre-condition:    Karel infront of RIGHT WALL (x,1) facing
EAST
         *                     behind him a line of beepers
         *
         * Post-condition:    Karel at 1:1 facing EAST, sitting on a
Stack of
         *                     Beepers representing the WIDTH of his
World
         */
        turnAround();
        while( frontIsClear() ) {
            while( beepersPresent() ) {
                pickBeeper();
                move();
                putBeeper();
                moveBack();
            }
            move();
        }
```

```
        turnAround();
    }

    private void moveBack() {
        /* moves karol ONE STEP back keeping direction heading
         *
         */
        turnAround();
        move();
        turnAround();
    }


    private void goMiddle() {
        /* this method moves karel (and the beepers left at his place)
to the next
         * position, taking 2 Beepers per Step.
         *
         * Pre-condition:    Karel at 1,1 facing EAST
         *                     X Beepers at 1,1
         *
         * Post-condition:    Karel at x,1 (near middle) facing EAST
         *                     no beepers left
         */

        pickBeeper(); // off by one bug: need to walk x-1 steps

        while( beepersPresent() ) {
            pickBeeper(); // take one beeper per Step
            pushStack();
            move();
            if( beepersPresent() ) { // Still Beepers left?
                pickBeeper(); // take one MORE beeper as charge/cost
            }
        }
    }



    private void pushStack() {
        /* this method moves the whole stack of beepers on karels
actual
         * position ONE STEP further keeping the direction.
         *
         * Pre-condition:    Karel at x,1 facing EAST with
         *                     X Beepers at his position.
         *
         * Post-condition:    Karel at x+1,1 facing EAST with
         *                     X-1 Beepers at his position.
         */
```

```
        while( beepersPresent() ) {
            pickBeeper(); // take one beeper as charge/cost
            move();
            putBeeper();
            moveBack();
        }
    }
}
```

— *Axel Werner* *2011-02-05 17:40*

java, karel, stanford, university, cs106, computer, science, learning, programming