```
package hw3;

import static api.Direction.*;

import api.Direction;
import api.Orientation;
/**
 * Represents a block in the Block Slider game.
 * @author Maxwell Skinner
 */
public class Block {
	/**
	 * The row coordinate of the block.
	 */
	private int firstRowCoordinate;
	/**
	 * The column coordinate of the block.
	 */
	private int firstColumnCoordinate;
	/**
	 * The length of the block.
	 */
	private int blockLength;
	/**
	 * The orientation of the block.
	 */
	private Orientation blockOrientation;
	/**
	 * The row coordinate of the block held for resetting
	 */
	private int firstRowCoordinateHold;
	/**
	 * The column coordinate of the block held for resetting.
	 */
	private int firstColumnCoordinateHold;
	/**
	 * Constructs a new Block with a specific location relative to the board. The
	 * upper/left most corner of the block is given as firstRow and firstCol. All
	 * blocks are only one cell wide. The length of the block is specified in
cells.
	 * The block can either be horizontal or vertical on the board as specified
by
	 * orientation.
	 *
	 * @param firstRow    the first row that contains the block
	 * @param firstCol    the first column that contains the block
	 * @param length      block length in cells
	 * @param orientation either HORIZONTAL or VERTICAL
	 */
	public Block(int firstRow, int firstCol, int length, Orientation orientation)
{
		firstRowCoordinate = firstRow;
		firstColumnCoordinate = firstCol;
		blockLength = length;
		blockOrientation = orientation;
		firstRowCoordinateHold = firstRow;
		firstColumnCoordinateHold = firstCol;
	}
```

```
	/**
	 * Resets the position of the block to the original firstRow and firstCol
values
	 * that were passed to the constructor during initialization of the the
block.
	 */
	public void reset() {
		firstRowCoordinate = firstRowCoordinateHold;
		firstColumnCoordinate = firstColumnCoordinateHold;
	}

	/**
	 * Move the blocks position by one cell in the direction specified. The
blocks
	 * first column and row should be updated. The method will only move VERTICAL
	 * blocks UP or DOWN and HORIZONTAL blocks RIGHT or LEFT. Invalid movements
are
	 * ignored.
	 *
	 * @param dir direction to move (UP, DOWN, RIGHT, or LEFT)
	 */
	public void move(Direction dir) {
		if(getOrientation() == Orientation.VERTICAL) {
			if(dir == Direction.UP) {
					firstRowCoordinate -= 1;
			}
			else if (dir == Direction.DOWN) {
					firstRowCoordinate += 1;

			}
		}
		else if(getOrientation()==Orientation.HORIZONTAL) {
			if(dir == Direction.RIGHT) {
					firstColumnCoordinate += 1;

			}
			else if(dir == Direction.LEFT) {
					firstColumnCoordinate -= 1;
			}
		}
	}

	/**
	 * Gets the first row of the block on the board.
	 *
	 * @return first row
	 */
	public int getFirstRow() {
		return firstRowCoordinate;
	}

	/**
	 * Sets the first row of the block on the board.
	 *
	 * @param firstRow first row
	 */
	public void setFirstRow(int firstRow) {
		firstRowCoordinate = firstRow;
	}
```

```
	/**
	 * Gets the first column of the block on the board.
	 *
	 * @return first column
	 */
	public int getFirstCol() {
		return firstColumnCoordinate;
	}

	/**
	 * Sets the first column of the block on the board.
	 *
	 * @param firstCol first column
	 */
	public void setFirstCol(int firstCol) {
		firstColumnCoordinate = firstCol;
	}

	/**
	 * Retrieves legnth of block
	 *
	 * @return length measured in cells
	 */
	public int getLength() {
		return blockLength;
	}

	/**
	 * Retrieves orientation of block
	 *
	 * @return either VERTICAL or HORIZONTAL
	 */
	public Orientation getOrientation() {
		return blockOrientation;
	}

	@Override
	public String toString() {
		return "(row=" + getFirstRow() + ", col=" + getFirstCol() + ", len=" +
getLength()
				+ ", ori=" + getOrientation() + ")";
	}
}
```