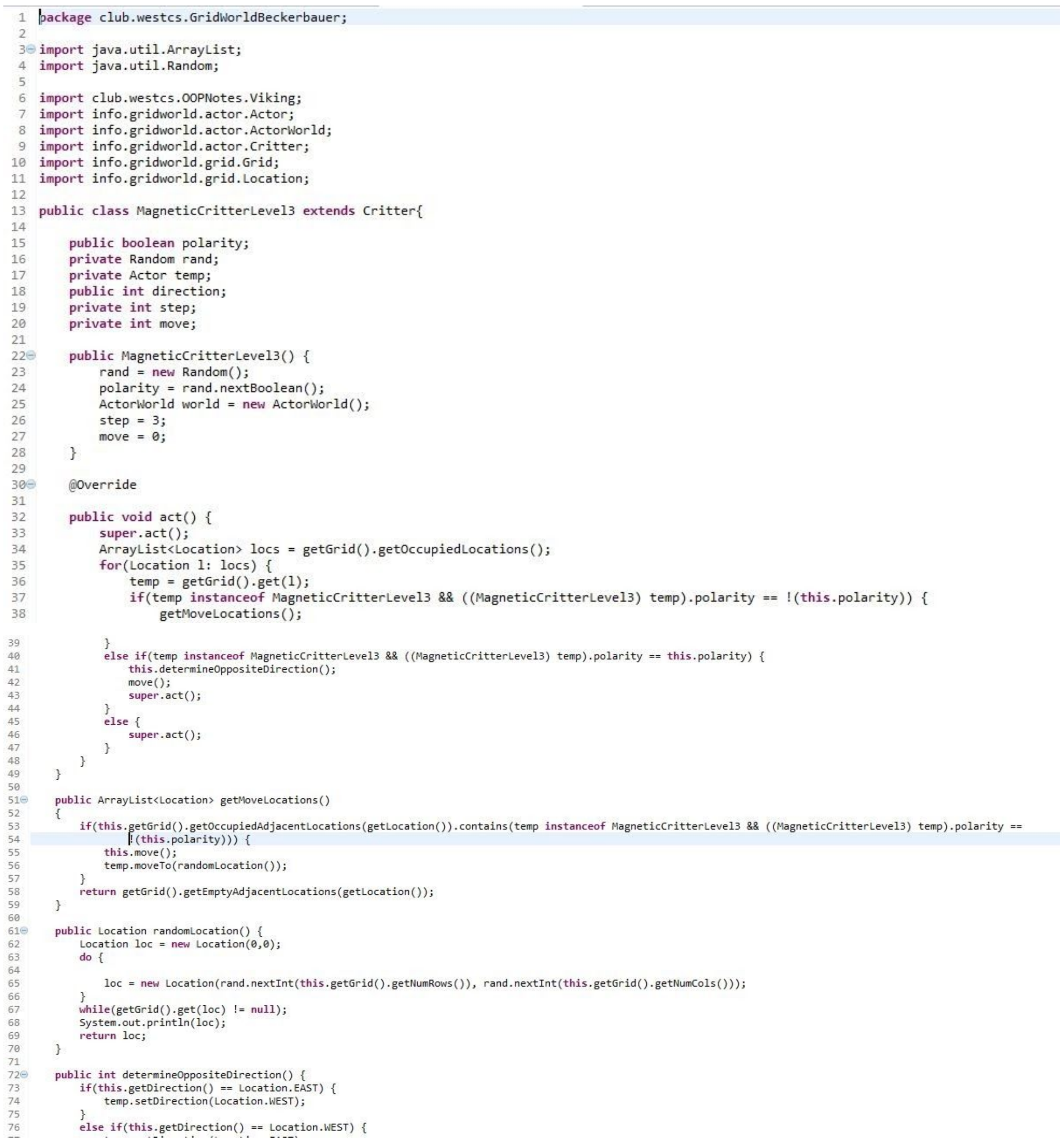
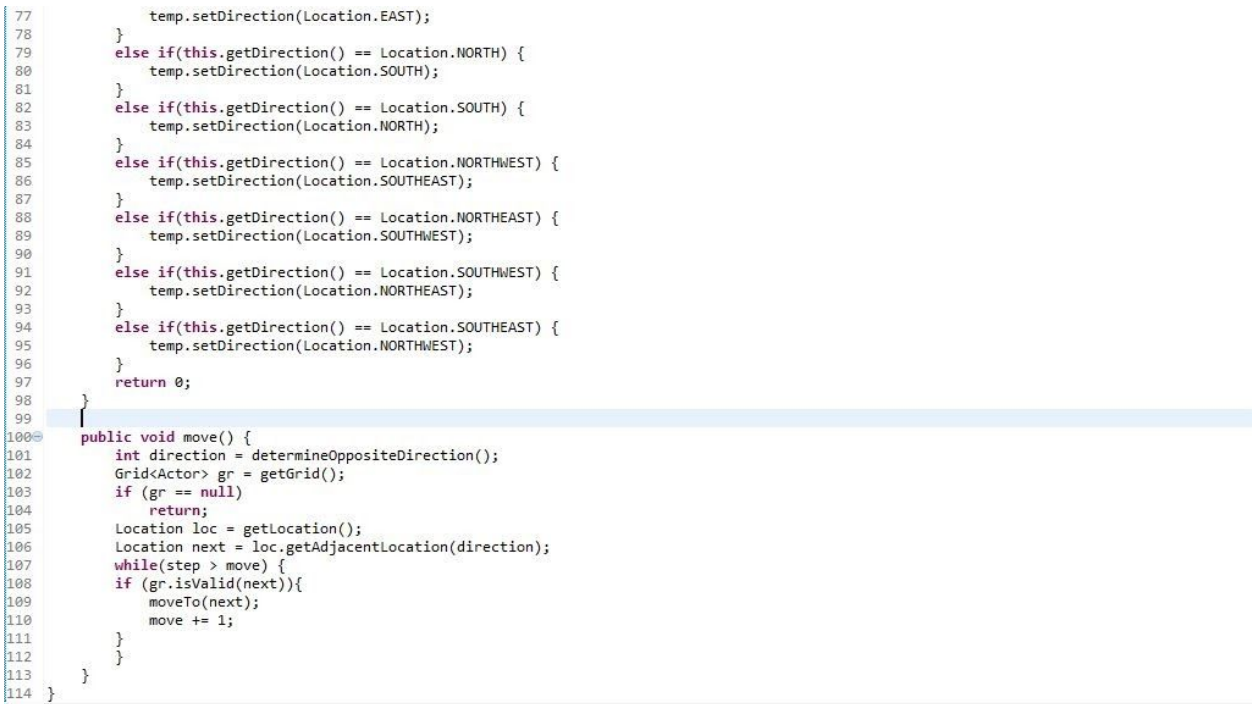


```java
package club.westcs.GridWorldBeckerbauer;

import java.util.ArrayList;
import java.util.Random;

import club.westcs.OOPNotes.Viking;
import info.gridworld.actor.Actor;
import info.gridworld.actor.ActorWorld;
import info.gridworld.actor.Critter;
import info.gridworld.grid.Grid;
import info.gridworld.grid.Location;

public class MagneticCritterLevel3 extends Critter{

    public boolean polarity;
    private Random rand;
    private Actor temp;
    public int direction;
    private int step;
    private int move;

    public MagneticCritterLevel3() {
        rand = new Random();
        polarity = rand.nextBoolean();
        ActorWorld world = new ActorWorld();
        step = 3;
        move = 0;
    }

    @Override

    public void act() {
        super.act();
        ArrayList<Location> locs = getGrid().getOccupiedLocations();
        for(Location l: locs) {
            temp = getGrid().get(l);
            if(temp instanceof MagneticCritterLevel3 && ((MagneticCritterLevel3) temp).polarity == !(this.polarity)) {
                getMoveLocations();

            }
            else if(temp instanceof MagneticCritterLevel3 && ((MagneticCritterLevel3) temp).polarity == this.polarity) {
                this.determineOppositeDirection();
                move();
                super.act();
            }
            else {
                super.act();
            }
        }
    }

    public ArrayList<Location> getMoveLocations()
    {
        if(this.getGrid().getOccupiedAdjacentLocations(getLocation()).contains(temp instanceof MagneticCritterLevel3 && ((MagneticCritterLevel3) temp).polarity ==
                !(this.polarity))) {
            this.move();
            temp.moveTo(randomLocation());
        }
        return getGrid().getEmptyAdjacentLocations(getLocation());
    }

    public Location randomLocation() {
        Location loc = new Location(0,0);
        do {

            loc = new Location(rand.nextInt(this.getGrid().getNumRows()), rand.nextInt(this.getGrid().getNumCols()));
        }
        while(getGrid().get(loc) != null);
        System.out.println(loc);
        return loc;
    }

    public int determineOppositeDirection() {
        if(this.getDirection() == Location.EAST) {
            temp.setDirection(Location.WEST);
        }
        else if(this.getDirection() == Location.WEST) {
```



```java
            temp.setDirection(Location.EAST);
        }
        else if(this.getDirection() == Location.NORTH) {
            temp.setDirection(Location.SOUTH);
        }
        else if(this.getDirection() == Location.SOUTH) {
            temp.setDirection(Location.NORTH);
        }
        else if(this.getDirection() == Location.NORTHWEST) {
            temp.setDirection(Location.SOUTHEAST);
        }
        else if(this.getDirection() == Location.NORTHEAST) {
            temp.setDirection(Location.SOUTHWEST);
        }
        else if(this.getDirection() == Location.SOUTHWEST) {
            temp.setDirection(Location.NORTHEAST);
        }
        else if(this.getDirection() == Location.SOUTHEAST) {
            temp.setDirection(Location.NORTHWEST);
        }
        return 0;
    }

    public void move() {
        int direction = determineOppositeDirection();
        Grid<Actor> gr = getGrid();
        if (gr == null)
            return;
        Location loc = getLocation();
        Location next = loc.getAdjacentLocation(direction);
        while(step > move) {
        if (gr.isValid(next)){
            moveTo(next);
            move += 1;
        }
        }
    }
}
```