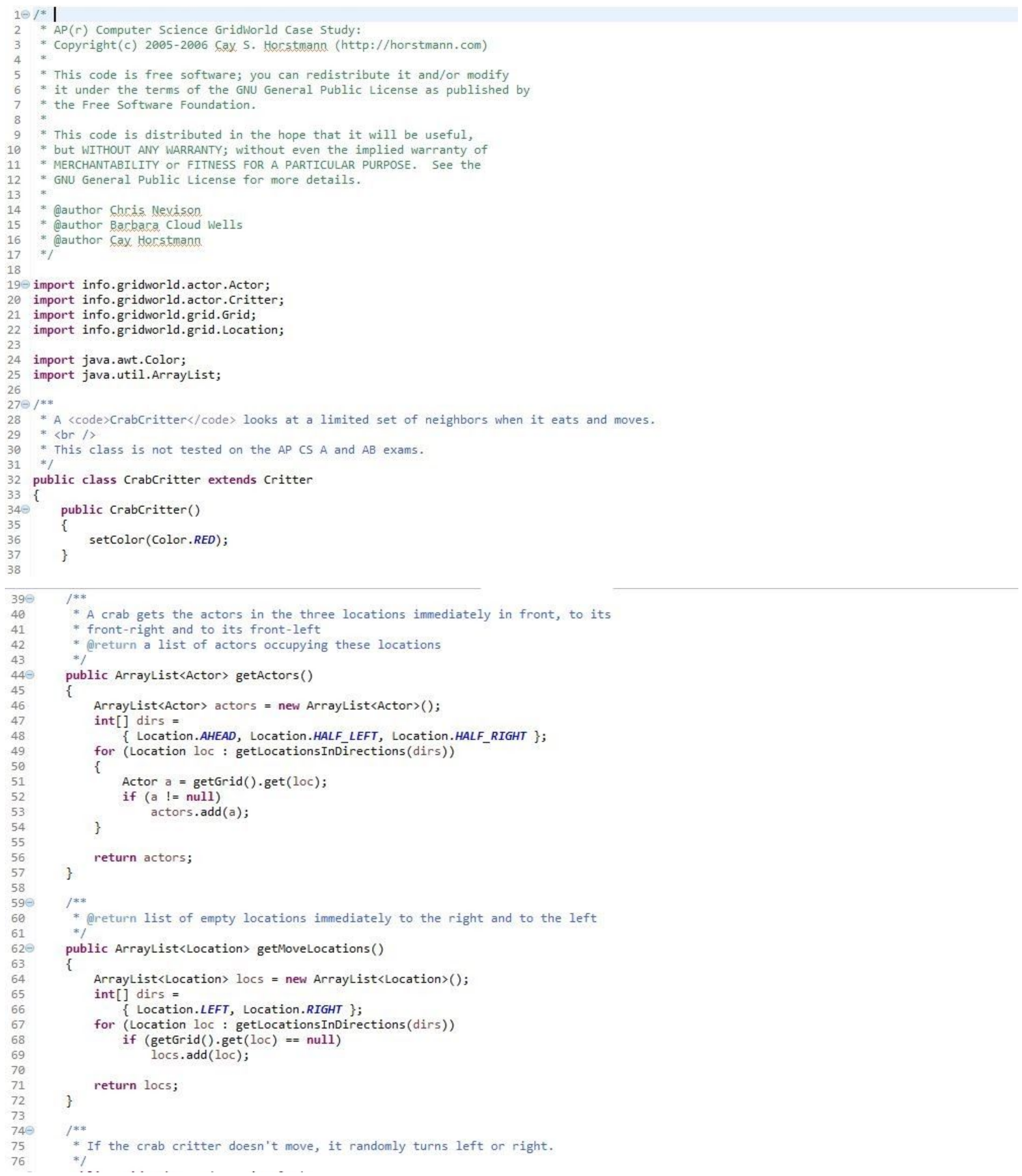
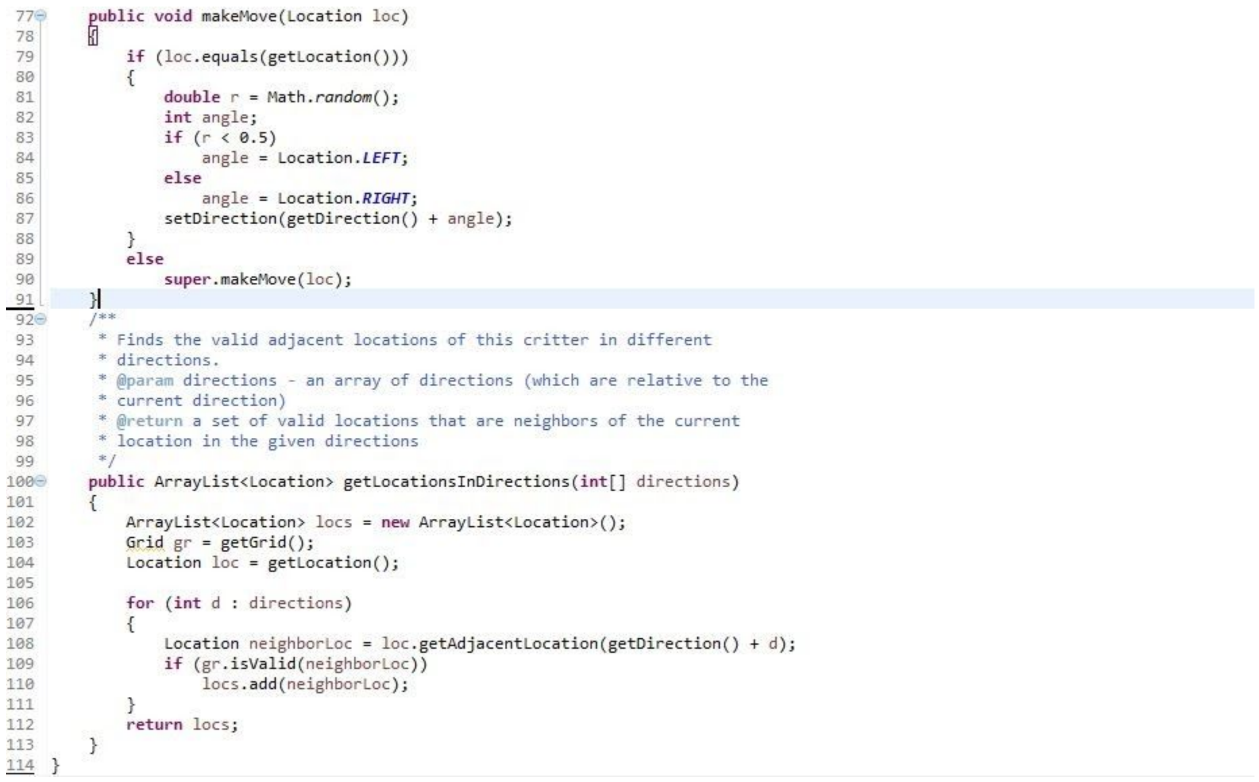


```java
/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2005-2006 Cay S. Horstmann (http://horstmann.com)
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * @author Chris Nevison
 * @author Barbara Cloud Wells
 * @author Cay Horstmann
 */

import info.gridworld.actor.Actor;
import info.gridworld.actor.Critter;
import info.gridworld.grid.Grid;
import info.gridworld.grid.Location;

import java.awt.Color;
import java.util.ArrayList;

/**
 * A <code>CrabCritter</code> looks at a limited set of neighbors when it eats and moves.
 * <br />
 * This class is not tested on the AP CS A and AB exams.
 */
public class CrabCritter extends Critter
{
    public CrabCritter()
    {
        setColor(Color.RED);
    }

    /**
     * A crab gets the actors in the three locations immediately in front, to its
     * front-right and to its front-left
     * @return a list of actors occupying these locations
     */
    public ArrayList<Actor> getActors()
    {
        ArrayList<Actor> actors = new ArrayList<Actor>();
        int[] dirs =
            { Location.AHEAD, Location.HALF_LEFT, Location.HALF_RIGHT };
        for (Location loc : getLocationsInDirections(dirs))
        {
            Actor a = getGrid().get(loc);
            if (a != null)
                actors.add(a);
        }

        return actors;
    }

    /**
     * @return list of empty locations immediately to the right and to the left
     */
    public ArrayList<Location> getMoveLocations()
    {
        ArrayList<Location> locs = new ArrayList<Location>();
        int[] dirs =
            { Location.LEFT, Location.RIGHT };
        for (Location loc : getLocationsInDirections(dirs))
            if (getGrid().get(loc) == null)
                locs.add(loc);

        return locs;
    }

    /**
     * If the crab critter doesn't move, it randomly turns left or right.
     */
```



```java
    public void makeMove(Location loc)
    {
        if (loc.equals(getLocation()))
        {
            double r = Math.random();
            int angle;
            if (r < 0.5)
                angle = Location.LEFT;
            else
                angle = Location.RIGHT;
            setDirection(getDirection() + angle);
        }
        else
            super.makeMove(loc);
    }
    /**
     * Finds the valid adjacent locations of this critter in different
     * directions.
     * @param directions - an array of directions (which are relative to the
     * current direction)
     * @return a set of valid locations that are neighbors of the current
     * location in the given directions
     */
    public ArrayList<Location> getLocationsInDirections(int[] directions)
    {
        ArrayList<Location> locs = new ArrayList<Location>();
        Grid gr = getGrid();
        Location loc = getLocation();

        for (int d : directions)
        {
            Location neighborLoc = loc.getAdjacentLocation(getDirection() + d);
            if (gr.isValid(neighborLoc))
                locs.add(neighborLoc);
        }
        return locs;
    }
}
```