实现一个瘟疫传播的可视化模拟

* 实验项目类型：设计性
*此表由学生按顺序填写

课程名称___面向对象程序设计/JAVA 语言___成绩评定________
实验项目名称_实现一个瘟疫传播的可视化模拟__
指导老师__干晓聪__
实验项目编号_1_实验项目类型__设计性__实验地点_数学系机房_
学生姓名___郭彦培___学号___2022101149___
学院_信息科学技术学院_系_数学系_专业_信息管理与信息系统_
实验时间_2023 年 11 月 1 日上午_~__2023 年 11 月 1 日中午__

一、 实验目的

在`JPanel`的基础上实现面板类`MyPanel`，练习`swing`库的使用，并且写出有实用价值的可视化程序。

二、 实验环境

计算机：PC X64

操作系统：Windows

编程语言：Java

IDE：IntelliJ IDEA

三、 程序原理

实现`Hospital`、`Persion`、`Person Poll`、`Bed`等类进行数据计算，随后传递给`myPanal`进行绘图。在`myPanal`中申请`Timer`进行数据的周期性重绘和刷新。

四、 程序代码

文件`sis9\Bed.java`实现了一个`Bed`类，用于计算医院床位相关数据

```java
package sis9;

public class Bed extends Point {
    public Bed(int x, int y) {
        super(x, y);
    }
    private boolean isEmpty=true;

    public boolean isEmpty() {
        return isEmpty;
    }

    public void setEmpty(boolean empty) {
        isEmpty = empty;
    }
}
```

文件`sis9\City.java`实现了一个`City`类，用于确定城市位置

```java
package sis9;

public class City {
    private int centerX;
    private int centerY;

    public City(int centerX, int centerY) {
        this.centerX = centerX;
        this.centerY = centerY;
    }

    public int getCenterX() {
        return centerX;
    }

    public void setCenterX(int centerX) {
        this.centerX = centerX;
    }

    public int getCenterY() {
        return centerY;
    }

    public void setCenterY(int centerY) {
        this.centerY = centerY;
    }
}
```

文件`sis9\Constants.java`实现了一个`Constants`类，用于存放常量

```java
package sis9;

public class Constants {

    public static int ORIGINAL_COUNT = 50;//初始感染数量
    public static float BROAD_RATE = 0.8f;//传播率
    public static float SHADOW_TIME = 140;//潜伏时间
    public static int HOSPITAL_RECEIVE_TIME = 10;//医院收治响应时间
    public static int BED_COUNT = 1000;//医院床位
    public static float u = -0.99f;//流动意向平均值
    public static int CITY_PERSON_SIZE = 5000;//城市总人口数量

}
```

文件`sis9\Hospital.java`实现了一个`Hospital`类，用于计算医院相关数据

```java
package sis9;

import java.util.ArrayList;
import java.util.List;

public class Hospital {

    private int x = 800;
    private int y = 110;

    private int width;
    private int height = 606;

    public int getWidth() {
        return width;
    }


    public int getHeight() {
        return height;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

    private static Hospital hospital = new Hospital();

    public static Hospital getInstance() {
        return hospital;
    }

    private Point point = new Point(800, 100);
    private List<Bed> beds = new ArrayList<>();

    private Hospital() {
        if (Constants.BED_COUNT == 0) {
            width = 0;
            height = 0;
        }
        int column = Constants.BED_COUNT / 100;
        width = column * 6;

        for (int i = 0; i < column; i++) {

            for (int j = 10; j <= 610; j += 6) {
                Bed bed = new Bed(point.getX() + i * 6, point.getY() + j);
                beds.add(bed);

            }

        }
    }

    public Bed pickBed() {
        for (Bed bed : beds) {
            if (bed.isEmpty()) {
                return bed;
            }
        }
        return null;
    }
}
```

文件`sis9\Main.java`实现了程序入口

```
package sis9;

import javax.swing.*;
import java.util.List;
import java.util.Random;

public class Main {
    public static void main(String[] args) {
        initPanel();
        initInfected();
    }

    private static void initPanel(){
        MyPanel p = new MyPanel();
        Thread panelThread = new Thread(p);
        JFrame frame = new JFrame();
        frame.add(p);
        frame.setSize(1000, 800);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
        frame.setTitle("瘟疫传播模拟");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        panelThread.start();
    }

    private static void initInfected() {
        List<Person> people = PersonPool.getInstance().getPersonList();
        for (int i = 0; i < Constants.ORIGINAL_COUNT; i++) {
            Person person;
            do {
                person = people.get(new Random().nextInt(people.size() -
1));
            } while (person.isInfected());
            person.beInfected();
        }
    }
}
```

文件`sis9\MoveTarget.java`实现了一个`MoveTarget`类，用于计算点云移动

```
package sis9;

public class MoveTarget {
    private int x;
    private int y;
    private boolean arrived=false;

    public MoveTarget(int x, int y) {
        this.x = x;
        this.y = y;
    }

    public int getX() {
        return x;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }

    public boolean isArrived() {
        return arrived;
    }

    public void setArrived(boolean arrived) {
        this.arrived = arrived;
    }
}
```

文件`sis9\MyPanel.java`实现了一个`MyPanel`类，用于绘图

```
package sis9;

import javax.swing.*;
import java.awt.*;
import java.util.List;
import java.util.Timer;
import java.util.TimerTask;

public class MyPanel extends JPanel implements Runnable {

    private int pIndex = 0;

    public MyPanel() {
        super();
        this.setBackground(new Color(0x444444));
    }

    @Override
    public void paint(Graphics g) {
        super.paint(g);
        g.setColor(new Color(0x00ff00));//设置医院边界颜色
        //绘制医院边界
        g.drawRect(Hospital.getInstance().getX(),
Hospital.getInstance().getY(),
                Hospital.getInstance().getWidth(),
Hospital.getInstance().getHeight());
        g.setFont(new Font("微软雅黑", Font.BOLD, 16));
        g.setColor(new Color(0x00ff00));
        g.drawString("医院", Hospital.getInstance().getX() +
Hospital.getInstance().getWidth() / 4, Hospital.getInstance().getY() - 16);

        List<Person> people = PersonPool.getInstance().getPersonList();
        if (people == null) {
            return;
        }
        people.get(pIndex).update();
        for (Person person : people) {
            switch (person.getState()) {
                case Person.State.NORMAL: {
                    g.setColor(new Color(0xdddddd));
                    break;
                }
                case Person.State.SHADOW: {
                    g.setColor(new Color(0xffee00));
                    break;
                }
                case Person.State.CONFIRMED:
                    g.setColor(new Color(0xff0000));
                    break;
                case Person.State.FREEZE: {
                    g.setColor(new Color(0x48FFFC));
                    break;
                }
            }
            person.update();
            g.fillOval(person.getX(), person.getY(), 3, 3);

        }
        pIndex++;
        if (pIndex >= people.size()) {
            pIndex = 0;
        }

        //显示数据信息
        g.setColor(Color.WHITE);
        g.drawString("城市总人数: " + Constants.CITY_PERSON_SIZE, 16, 40);
        g.setColor(new Color(0xdddddd));
        g.drawString("健康者人数: " +
PersonPool.getInstance().getPeopleSize(Person.State.NORMAL), 16, 64);
        g.setColor(new Color(0xffee00));
        g.drawString("潜伏者人数: " +
PersonPool.getInstance().getPeopleSize(Person.State.SHADOW), 16, 88);
        g.setColor(new Color(0xff0000));
        g.drawString("感染者人数: " +
PersonPool.getInstance().getPeopleSize(Person.State.CONFIRMED), 16, 112);
        g.setColor(new Color(0x48FFFC));
        g.drawString("已隔离人数: " +
PersonPool.getInstance().getPeopleSize(Person.State.FREEZE), 16, 136);
        g.setColor(new Color(0x00ff00));
        g.drawString("空余病床: " + (Constants.BED_COUNT -
PersonPool.getInstance().getPeopleSize(Person.State.FREEZE)), 16, 160);
    }

    public static int worldTime = 0;

    public Timer timer = new Timer();

    class MyTimerTask extends TimerTask {
        @Override
        public void run() {
            MyPanel.this.repaint();
            worldTime++;
        }
    }

    @Override
    public void run() {
        timer.schedule(new MyTimerTask(), 0, 100);
    }


}
```

文件`sis9\Person.java`实现了一个`Person`类，用于计算人相关数据

```
package sis9;

import java.util.List;
import java.util.Random;


public class Person {
    private City city;
    private int x;
    private int y;
    private MoveTarget moveTarget;
    int sig = 1;


    double targetXU;
    double targetYU;
    double targetSig = 50;


    public interface State {//市民状态
        int NORMAL = 0;//未被感染
        int SHADOW = NORMAL + 1;//潜伏者

        int CONFIRMED = SHADOW + 1;//感染者
        int FREEZE = CONFIRMED + 1;//已隔离
    }
```

```java
public Person(City city, int x, int y) {
    this.city = city;
    this.x = x;
    this.y = y;
    targetXU = 100 * new Random().nextGaussian() + x;
    targetYU = 100 * new Random().nextGaussian() + y;

}

public boolean wantMove() {
    double value = sig * new Random().nextGaussian() + Constants.u;
    return value > 0;
}

private int state = State.NORMAL;

public int getState() {
    return state;
}

public void setState(int state) {
    this.state = state;
}

public int getX() {
    return x;
}

public void setX(int x) {
    this.x = x;
}

public int getY() {
    return y;
}

public void setY(int y) {
    this.y = y;
}

int infectedTime = 0;
int confirmedTime = 0;
```

```
    public boolean isInfected() {
        return state >= State.SHADOW;
    }

    public void beInfected() {
        state = State.SHADOW;
        infectedTime = MyPanel.worldTime;
    }

    public double distance(Person person) {
        return Math.sqrt(Math.pow(x - person.getX(), 2) + Math.pow(y -
person.getY(), 2));
    }

    private void freezy() {
        state = State.FREEZE;
    }

    private void moveTo(int x, int y) {
        this.x += x;
        this.y += y;
    }

    private void action() {
        if (state == State.FREEZE) {
            return;
        }
        if (!wantMove()) {
            return;
        }
        if (moveTarget == null || moveTarget.isArrived()) {

            double targetX = targetSig * new Random().nextGaussian() +
targetXU;
            double targetY = targetSig * new Random().nextGaussian() +
targetYU;
            moveTarget = new MoveTarget((int) targetX, (int) targetY);

        }


        int dX = moveTarget.getX() - x;
        int dY = moveTarget.getY() - y;
        double length = Math.sqrt(Math.pow(dX, 2) + Math.pow(dY, 2));
        if (length < 1) {
            moveTarget.setArrived(true);
            return;
        }
        int udX = (int) (dX / length);
        if (udX == 0 && dX != 0) {
            if (dX > 0) {
                udX = 1;
            } else {
                udX = -1;
            }
        }
        int udY = (int) (dY / length);
        if (udY == 0 && udY != 0) {
            if (dY > 0) {
                udY = 1;
            } else {
                udY = -1;
            }
        }

        if (x > 700) {
            moveTarget = null;
            if (udX > 0) {
                udX = -udX;
            }
        }
        moveTo(udX, udY);
    }

    private float SAFE_DIST = 2f;

    public void update() {
        if (state >= State.FREEZE) {
            return;
        }
        if (state == State.CONFIRMED && MyPanel.worldTime - confirmedTime
>= Constants.HOSPITAL_RECEIVE_TIME) {
            Bed bed = Hospital.getInstance().pickBed();
            if (bed == null) {
            } else {
                state = State.FREEZE;
                x = bed.getX();
```

```java
                y = bed.getY();
                bed.setEmpty(false);
            }
        }
        if (MyPanel.worldTime - infectedTime > Constants.SHADOW_TIME &&
state == State.SHADOW) {
            state = State.CONFIRMED;
            confirmedTime = MyPanel.worldTime;
        }

        action();

        List<Person> people = PersonPool.getInstance().personList;
        if (state >= State.SHADOW) {
            return;
        }
        for (Person person : people) {
            if (person.getState() == State.NORMAL) {
                continue;
            }
            float random = new Random().nextFloat();
            if (random < Constants.BROAD_RATE && distance(person) <
SAFE_DIST) {
                this.beInfected();
            }
        }
    }
}
```

文件`sis9\PersonPool.java`实现了一个`PersonPool`类，用于计算人口池相关数据

```java
package sis9;

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

public class PersonPool {
    private static PersonPool personPool = new PersonPool();

    public static PersonPool getInstance() {
        return personPool;
    }
    List<Person> personList = new ArrayList<Person>();

    public List<Person> getPersonList() {
        return personList;
    }


    public int getPeopleSize(int state) {
        if (state == -1) {
            return Constants.CITY_PERSON_SIZE;
        }
        int i = 0;
        for (Person person : personList) {
            if (person.getState() == state) {
                i++;
            }
        }
        return i;
    }

    private PersonPool() {
        City city = new City(400, 400);
        for (int i = 0; i < Constants.CITY_PERSON_SIZE; i++) {
            Random random = new Random();
            int x = (int) (100 * random.nextGaussian() +
city.getCenterX());
            int y = (int) (100 * random.nextGaussian() +
city.getCenterY());
            if (x > 700) {
                x = 700;
            }
            personList.add(new Person(city, x, y));
        }
    }
}
```

文件`sis9\Point.java`实现了一个`Point`类，用于计算点数据

```
package sis9;

public class Point {
    private int x;
```

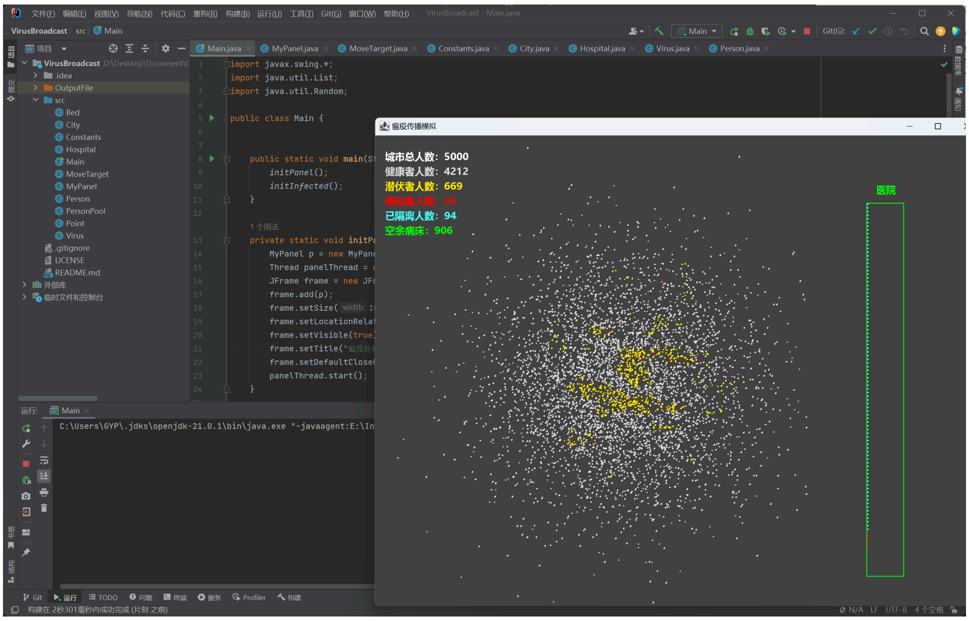
```java
    private int y;

    public Point(int x, int y) {
        this.x = x;
        this.y = y;
    }

    public int getX() {
        return x;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }
}
```

五、 出现的问题、原因与解决方法

编码过程中大量参考`JPanel`的`reference`，并且结合一些开源项目的实例，因此非常顺利，没有出现什么问题。

六、 测试数据与运行结果

刚开始模拟



```
import javax.swing.*;
import java.util.List;
import java.util.Random;

public class Main {


    public static void main(St
        initPanel();
        initInfected();
    }

    1 个用法
    private static void initPa
        MyPanel p = new MyPane
        Thread panelThread = n
        JFrame frame = new JFr
        frame.add(p);
        frame.setSize( width: 1(
        frame.setLocationRelat
        frame.setVisible(true)
        frame.setTitle("瘟疫传播
        frame.setDefaultClose(
        panelThread.start();
    }
```

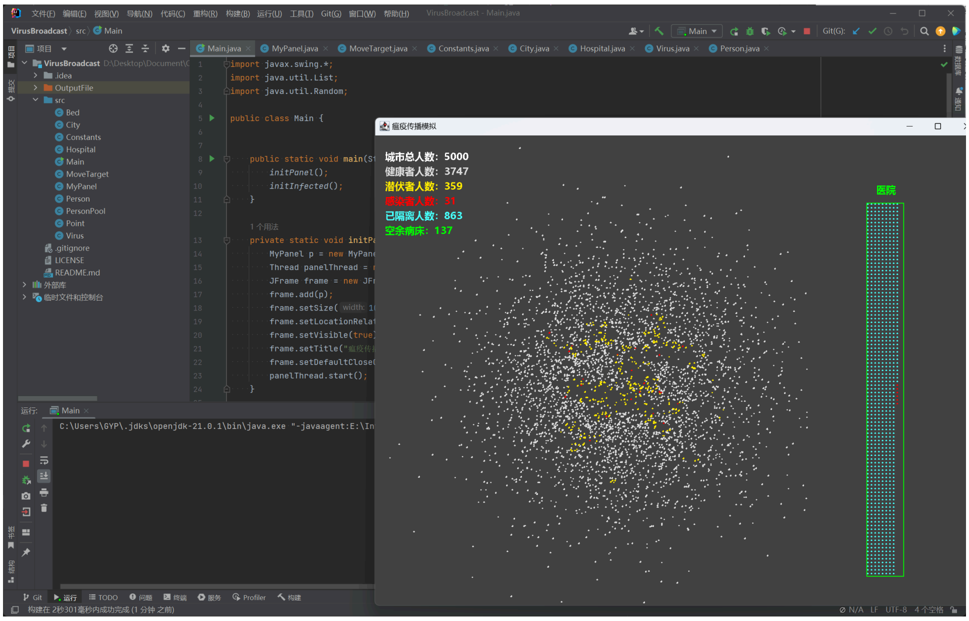
```
C:\Users\GYP\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:E:\In
```

瘟疫传播模拟

城市总人数：5000
健康者人数：4212
潜伏者人数：669
感染者人数：25
已隔离人数：94
空余病床：906

构建在 2秒301毫秒内成功完成 (片刻 之前)

数十秒后，医院接近满员



```
import javax.swing.*;
import java.util.List;
import java.util.Random;

public class Main {


    public static void main(St
        initPanel();
        initInfected();
    }

    1 个用法
    private static void initPa
        MyPanel p = new MyPane
        Thread panelThread = n
        JFrame frame = new JFr
        frame.add(p);
        frame.setSize( width: 1(
        frame.setLocationRelat
        frame.setVisible(true)
        frame.setTitle("瘟疫传播
        frame.setDefaultClose(
        panelThread.start();
    }
```

```
C:\Users\GYP\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:E:\In
```

瘟疫传播模拟

城市总人数：5000
健康者人数：3747
潜伏者人数：359
感染者人数：31
已隔离人数：863
空余病床：137

构建在 2秒301毫秒内成功完成 (1 分钟 之前)