Java 语言与系统设计课程实验报告

学生姓名	孔德彬
学生学号	8208181404
指导教师	郭克华
专业班级	计科 1801
完成日期	2019.11.18

计算机学院

目　录

实验一

一、目的与要求

1、自行下载最喜爱的小说 1 部。存到服务器中，格式自定。一般存储为文本文档。要求长篇小说，20 万字以上。举例说明：下载《三国演义》保存在服务器端。

2、该软件支持从服务器载入小说，对小说中的文本进行分析。举例说明：服务器端保存《三国演义》，客户端进行分析。

3、首先运行服务器。服务器运行之后，可以连接 1 个客户端。

4、运行客户端。用户能够输入昵称，确定，则连接到服务器。连接成功，即可出现功能界面。

客户端功能界面如下：

1、功能 1：载入小说。能够选择服务器端的小说。举例说明：客户端点击按钮，选择服务器端的文件名，《三国演义》传输到客户端。

2、功能 2：任意设置 10 个人姓名（可以预设置在客户端界面上），将这 10 个人在小说中的存在感进行排名，用柱状图表示。如何计算存在感？自己定义。点击按钮，存在感排名的柱状图可以保存到服务器端。举例说明：界面上设置“刘备、曹操、张飞、关羽、赵云、诸葛亮、吕布、貂蝉、董卓、孙权”，点击按钮，出现一个柱状图，显示存在感排名为：刘备、曹操、张飞、关羽、诸葛亮、赵云、孙权、吕布、董卓、貂蝉（只是举例说明）。

3、每个人在小说中活跃的位置是不一样的。任意输入 10 人中的 1 人，显示他在小说中出现的密度，画出密度图。建议用颜色深浅表示密度。点击按钮，密度图可以保存到服务器端。举例说明：输入“刘备”，在小说中前面部分密度大，后

面部分密度小。

4、如果两人在相距较短的一段文字中出现，我们认为两人有关系，距离越短，关系越近；如果多次在较短篇幅出现，则关系更近。对这 10 个人，根据他们关系的紧密程度进行归类，看看可以归为哪几类？并用图形形式直观表示。如何定义关系紧密程度？自己定义。点击按钮，紧密程度归类的图像内容，可以保存到服务器端。举例说明：这 10 个人，自动分为“刘备、张飞、关羽、赵云、诸葛亮”以及“曹操”、“吕布、董卓、貂蝉”、“孙权”（只是举例）。

5、附加题，不检查，做了不加分不减分。任意输入一个人，显示他最恨谁，最喜欢谁，最喜欢吃什么，经常去哪里活动，杀人多不多，是好人还是坏人？看与你的直觉是否符合？如果不太符合，说明可能的原因。

二、操作环境

硬件： PC

软件：eclipse

三、实验内容

1 server 包

1.1 Server.java：

主要的服务器功能，为客户端打开连接端口，并实现向客户端传输数据和接受客户端的数据传输功能。

2 client 包

2.1 Client.java:

主要是客户端主函数，负责与服务器进行连接，有一个文本框，可以输入使用者姓名，输入之后点击连接即可连接到服务器，跳转到主功能界面。

3 myframe 包

3.1 Fun1Frame.java:

功能一界面，主要负责完成功能一的下载小说功能。有两个按键，下载小说和返回按键。当点击下载小说后，客户端向服务器传输指定数值，服务器接收后开始向客户端传输小说内容，客户端接受服务端传来的小说内容，并将其贮存在指定的文件夹下。点击返回按键即可返回主功能界面。

3.2 Fun2Frame.java:

功能二界面，主要负责完成功能二以柱状图的形式反应人物出现的频率大小的功能。有两个按键，生成柱状图和返回按键。当点击生成柱状图按键时，客户端向服务器传输指定参数，服务器接收后调用 name 包里的函数对小说进行分析，之后生成相应的柱状图。点击返回按键返回主功能选择界面。

3.3 Fun3Frame.java:

功能三界面，主要负责完成功能三按照每章出现次数来生成密度图。在界面上预设好了十个人的名字以及对应的序号，在一个文本输入框中输入你想查询的人名对应的序号，然后点击保存密度图，客户端向服务器传输对应的参数，服务器接收到参数以后调用 name 包，生成对应的密度图。

3.4 Fun4Frame.java:

功能四界面，主要是负责完成功能四对预设好的十个人进行分类的功能。有两个

按钮，保存归类内容和返回按键。点击保存归类内容以后，客户端向服务端传入指定参数，服务端接收以后调用 name 包显示并保存分类内容。点击返回按键返回主功能选择界面。

3.5 MainFrame.java:

主功能选择界面，主要是负责功能的选择。当 client 连接成功后会跳转到主功能选择界面。有四个按键，每个按键对应着不同的四个功能，点击之后即可跳转到指定的功能。

4 name 包

4.1 Name.java:

对文本进行分析的主函数。预设有十个人的姓名，在 Name.java 里给又设置了一个新的 MainName 类，用来存放每个人的主要信息，并用 map 函数进行贮存。其中存放的主要信息有姓名，每章出现次数，总共出现次数等。对文本的每章进行扫描，并贮存每个人每章出现的次数，结束后，采用冒泡排序对 map 进行排序。当调用它时生成不同的图表。

4.2 BarChart.java:

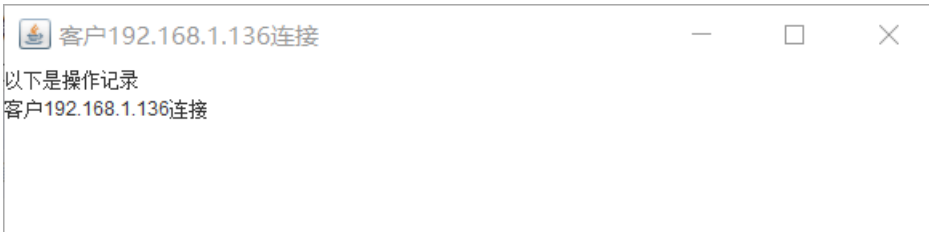
主要是用来生成柱状图的图表。Name.java 调用该类来生成对应的柱状图，由于功能二和功能三均需要柱状图，则通过改变构造函数参数的不同来区分要生成功能几的柱状图。生成柱状图的方法主要是调用外引入的包 JFreeChart。

4.3 PieChart.java:

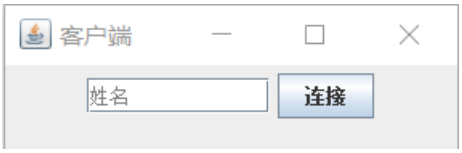
主要是生成对应的密度图，并通过他们每章出现次数的多少来设置对应章节区域的颜色深浅，生成饼状图的方法也是调用外引入的包 JFreeChart 来实现。

四、实验结果

1　客户端连接成功界面



2　客户端主界面

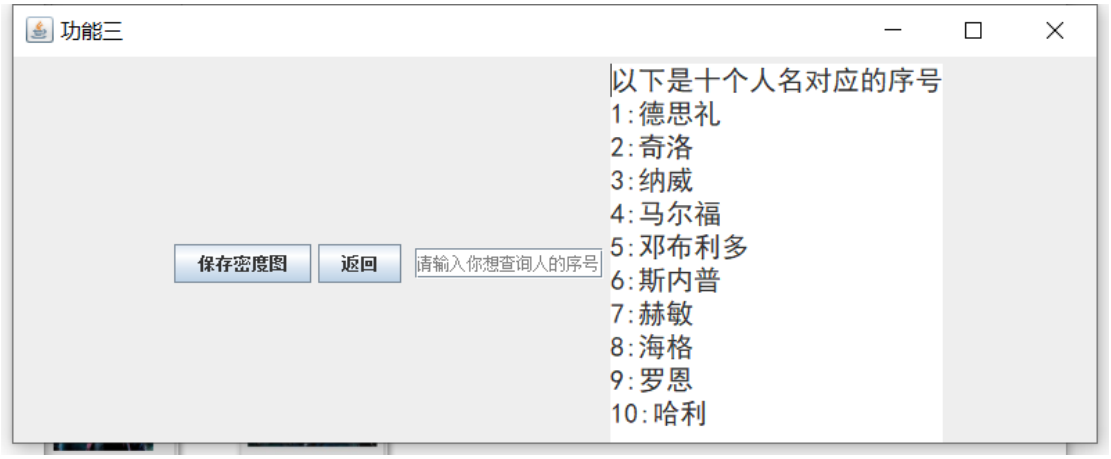


3　MainFrame 界面

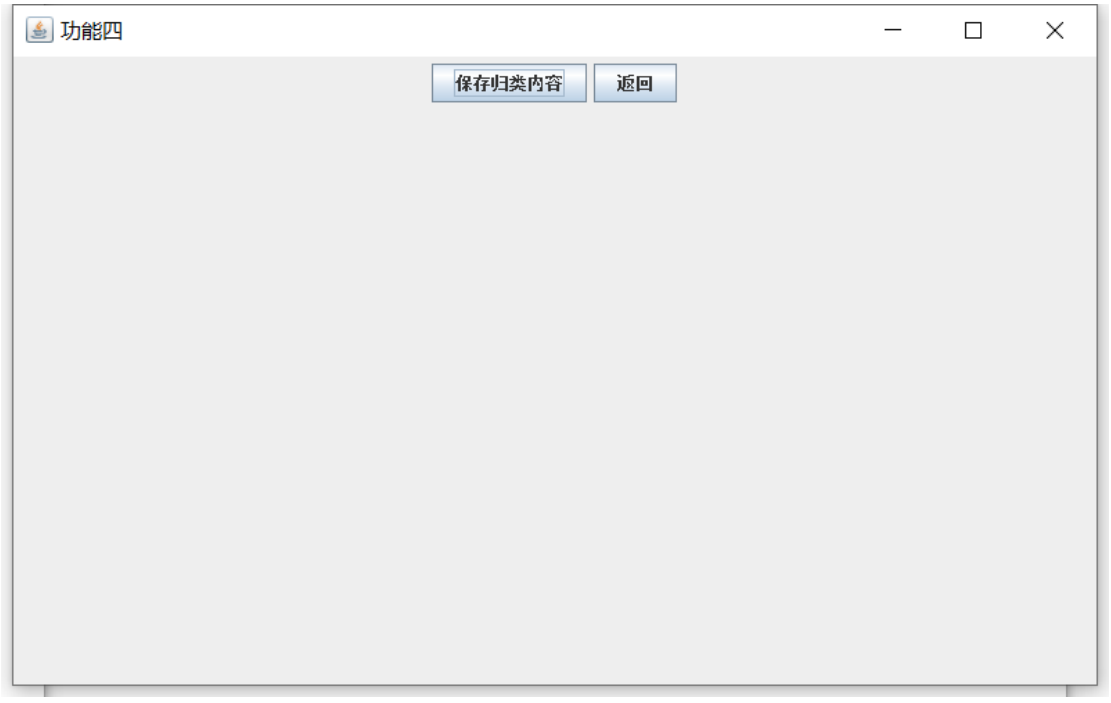
4 Fun1Frame 界面

5 Fun2Frame 界面

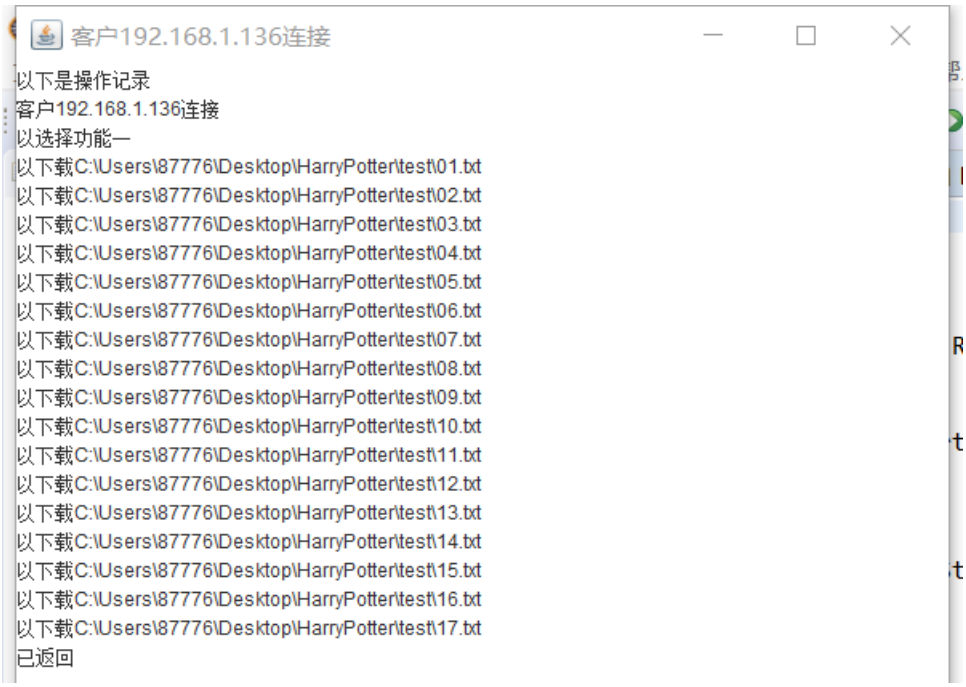
6 Fun3Frame 界面



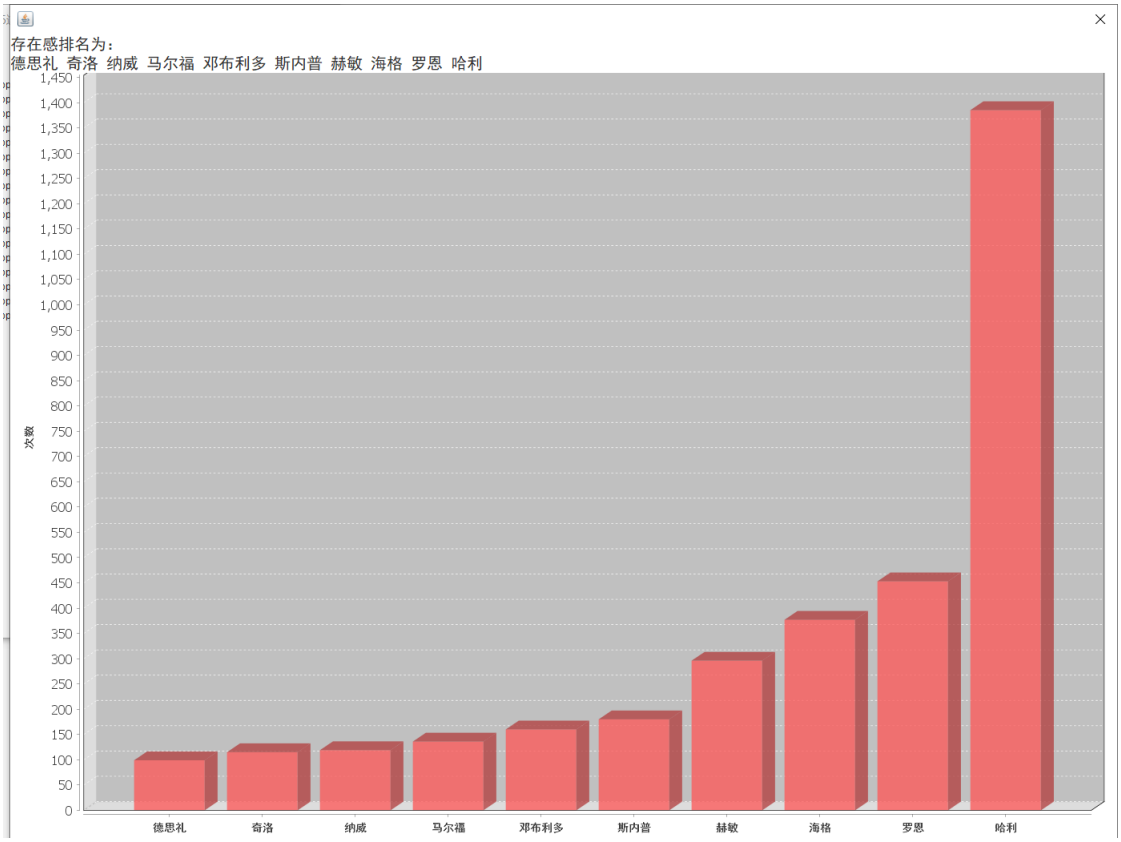
7 Fun4Frame 界面



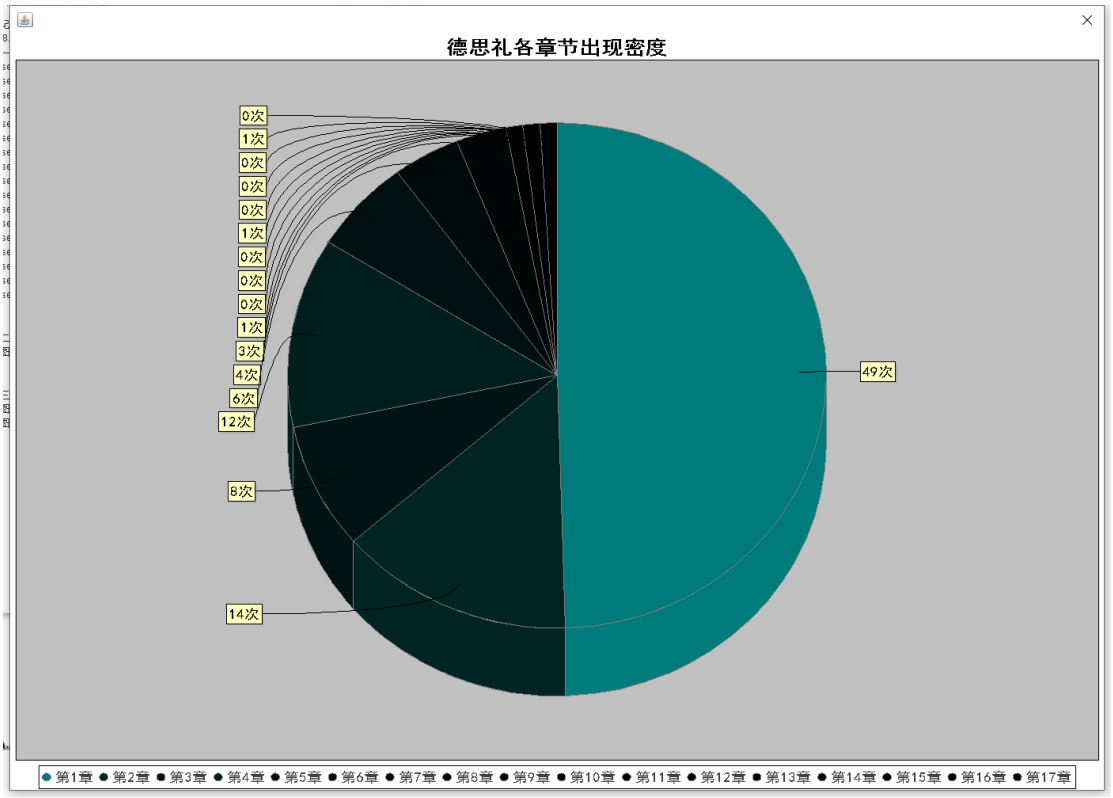
8　功能一下载完成服务器提示界面



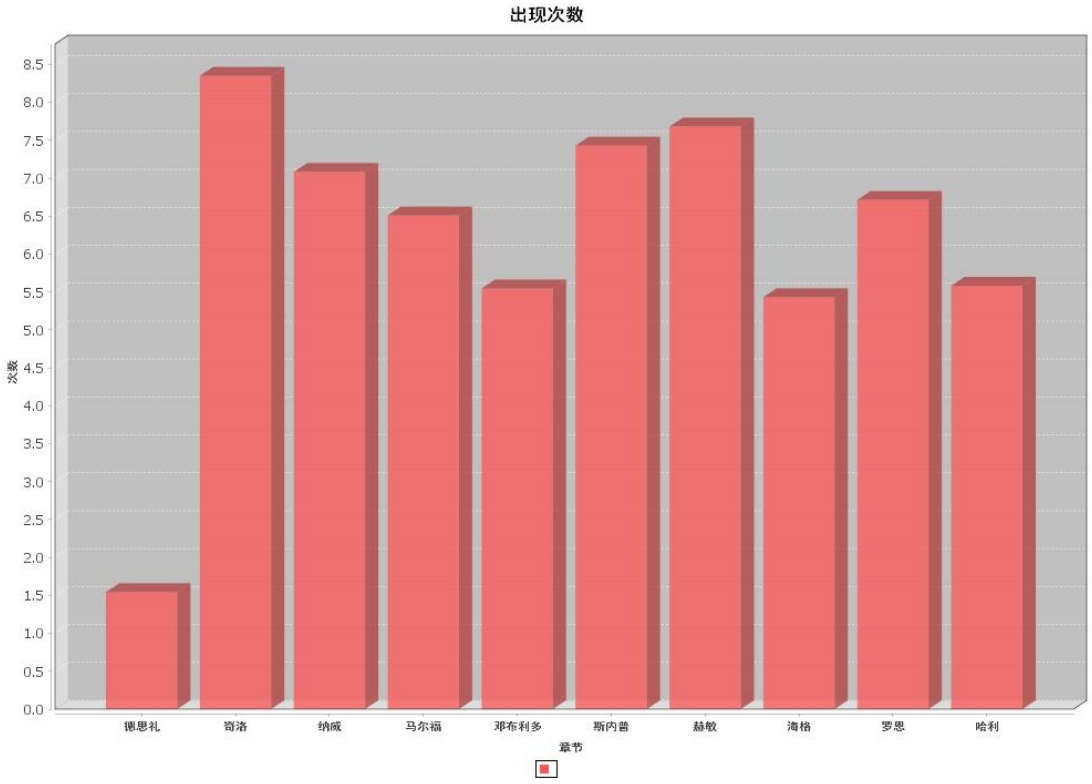
9　功能二生成柱状图界面



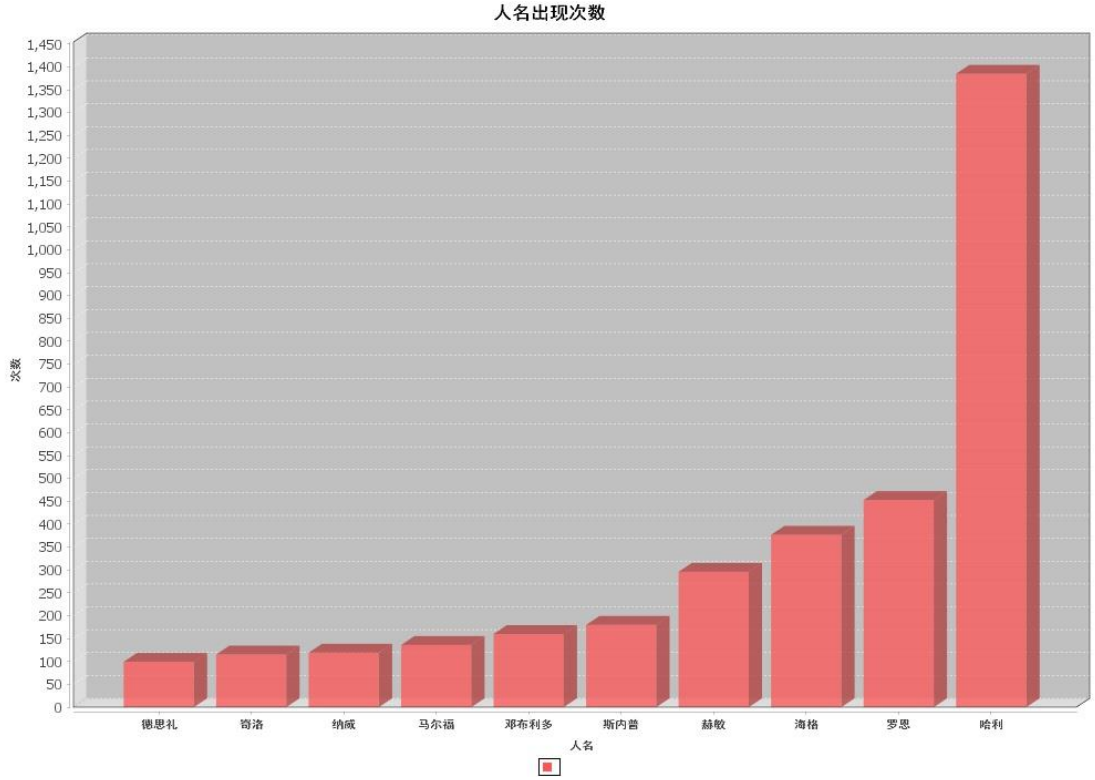
10　功能三生成密度图界面



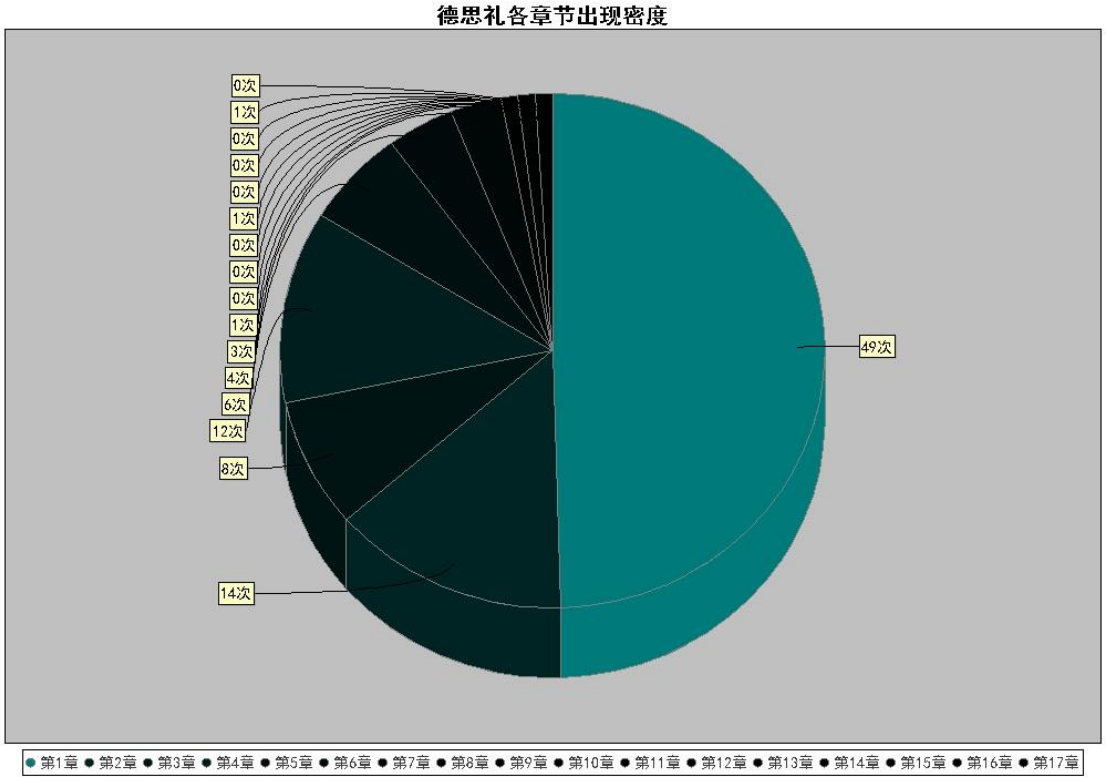
11　功能四进行分类界面



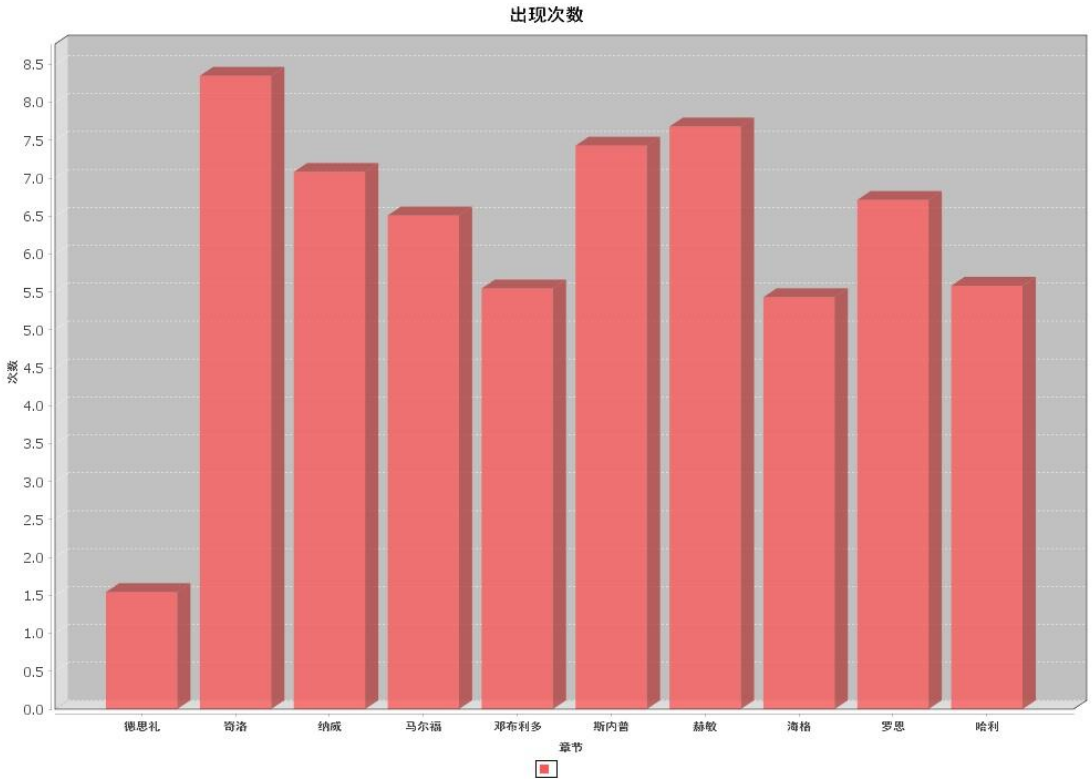
12　功能二生成柱状图



13　功能三生成密度图



14　功能四生成柱状图


出现次数
8.5
8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
次数
德思礼
奇洛
纳威
马尔福
邓布利多
斯内普
赫敏
海格
罗恩
哈利
章节

五、实验源代码

1. server 包

a) Server.java：

```
public class Server extends JFrame implements Runnable{

    private Socket s = null;
    private ServerSocket ss = null;
    private JTextArea taMsg = new JTextArea("以下是操作记录\n");

    public Server(){
        this.setTitle("服务端");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setBackground(Color.yellow);
        this.setSize(600,800);
        this.setVisible(true);
        this.add(taMsg,BorderLayout.CENTER);

        try {
            //监听9999端口
            ss = new ServerSocket(9999); //服务器开端口
            s = ss.accept();
            String clientAddress = s.getInetAddress().getHostAddress();
            this.setTitle("客户" + clientAddress + "连接");
            taMsg.append("客户" + clientAddress + "连接\n");
            new Thread(this).start();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }

    public void run() {
        OutputStream out = null;
        InputStream in = null;
        PrintWriter bufw = null;
        BufferedReader bufr = null;

        try {
            //生成名字对象
            Name n = new Name();

            while (true) {
                out = s.getOutputStream();              //向客户端传输数据
                in = s.getInputStream();                //接受客户端数据
```

```
                //写传出数据
                bufw = new PrintWriter(out,true);
                //读取传入数据
                bufr = new BufferedReader(new InputStreamReader(in));

                out.flush();

                String flag = bufr.readLine();
                System.out.println(flag);

                switch (Integer.valueOf(flag)) {

                case 0:
                    taMsg.append("已返回\n");
                    break;

                case 1:
                    taMsg.append("以选择功能一\n");
                    break;

                case 2:
                    taMsg.append("以选择功能二\n");
                    break;

                case 3:
                    taMsg.append("以选择功能三\n");
                    break;

                case 4:
                    taMsg.append("以选择功能四\n");
                    break;

                case 11:
                    File file1 = new File(
                            "C:\\Users\\87776\\Desktop\\HarryPotter\\test");
                    File[] fl = file1.listFiles();
                    for(File ft : fl) {
                        FileInputStream fis = new FileInputStream(ft);
                        InputStreamReader isr = new InputStreamReader(fis,
"utf-8");
                        BufferedReader br = new BufferedReader(isr);
                        String line = null;

                        while((line = br.readLine()) != null) {
```

```
                        if(!bufw.checkError()) {
                            bufw.flush();
                            System.out.println(line + "\r\n");
                            bufw.println(line + "\r\n");
                        }
                    }
                    taMsg.append("以下载" + ft + "\n");
                    fis.close();

                    bufw.flush();
                }
                bufw.flush();
                bufw.flush();
                break;

            case 22:
                n.printSum();
                n.getBarChart();

                taMsg.append("已生成柱状图\n");
                break;

            case 44:
                n.printAvgNum();
                n.getBarChart(0);
                break;

            default:
                n.print(Integer.valueOf(flag) - 31);
                taMsg.append("已生成密度图\n");

                break;
            }
        }
    } catch (IOException e) {
        System.out.println(e.getMessage());
    }
}

public static void main(String[] args) {
    new Server();
}
}
```

2. client 包

a) Client.java:

```
public class Client extends JFrame implements ActionListener, FocusListener{
    private Socket s = null;
    private JButton btConnect = new JButton("连接");
    private JTextField tfMsg  = new JTextField(10);
    public Client() {

        tfMsg.setText("姓名");
        tfMsg.setForeground(Color.GRAY);

        this.setLayout(new FlowLayout());
        this.setTitle("客户端");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.add(tfMsg);
        this.add(btConnect,BorderLayout.CENTER);
        this.setVisible(true);
        this.setSize(300, 100);
        this.setLocationRelativeTo(null);

        tfMsg.addFocusListener(this);
        btConnect.addActionListener(this);
    }

    @Override
    public void actionPerformed(ActionEvent e) {

        try {
//          s = new Socket("192.168.43.52", 9999);
            s = new Socket("192.168.43.52", 9999);
            this.setTitle("连接成功");
            new MainFrame(s);
            this.dispose();
        } catch (Exception e1) {
            System.out.println(e1.getMessage());
        }

    }

    @Override
    public void focusGained(FocusEvent e) {
        //当获取焦点时
        String temp = tfMsg.getText();
```

```
        if(temp.equals("姓名")) {
            tfMsg.setText("");
            tfMsg.setForeground(Color.black);
        }
    }

    @Override
    public void focusLost(FocusEvent e) {
        //当失去焦点时
        String temp = tfMsg.getText();
        if(temp.equals("")) {
            tfMsg.setForeground(Color.GRAY);
            tfMsg.setText("姓名");
        }
    }

    public static void main(String[] args) {
        new Client();
    }

}
```

3. myframe 包

a) Fun1Frame.java:

```
public class Fun1Frame extends JFrame implements ActionListener,Runnable{
    private Socket s;

    private JButton fun1Button = new JButton("下载小说");
    private JButton mainButton = new JButton("返回");
    private JPanel  fun1Panel  = new JPanel();

    public Fun1Frame(Socket s) {
        this.s = s;

        //绑定
        fun1Button.addActionListener(this);
        mainButton.addActionListener(this);
        fun1Panel.add(fun1Button);
        fun1Panel.add(mainButton);

        this.setTitle("功能一");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.add(fun1Panel,BorderLayout.NORTH);
```

```
        this.add(new JLabel(new
ImageIcon("C:\\Users\\87776\\Desktop\\download.jpg")));
        this.setSize(450,500);
        this.setVisible(true);
        this.setLocationRelativeTo(null);
    }

    public void actionPerformed(ActionEvent e) {

        Thread t = new Thread(this);

        if(e.getSource() == fun1Button) {
            try {
                OutputStream out = s.getOutputStream();           //获取服务端
的输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);

                bufw.println(11);                                 //发送数据给服
务端
                this.setTitle("正在下载小说...");
                t.start();
                this.setTitle("小说下载完成");
                bufw.println(0);
            } catch (Exception e1) {
                System.out.println(e1.getMessage());
            }
        }else {
            //返回主页面
            try {
                OutputStream out = s.getOutputStream();          //获取服务端的
输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);
                bufw.println(0);                                 //发送数据给服
务端
                new MainFrame(s);
                t.interrupt();
                this.dispose();
            } catch (Exception e2) {
                System.out.println(e2.getMessage());
            }
        }
    }

    public void run() {
```

```
        while(true) {
            try {
                FileOutputStream fos = new
FileOutputStream("C:\\Users\\87776\\Desktop\\客户端\\HarryPotter.txt");
                OutputStreamWriter osw = new OutputStreamWriter(fos, "utf-8");
                InputStream is = s.getInputStream();      //获取服务端的输入流，
为了获取服务端数据

                InputStreamReader isr = new InputStreamReader(is, "GB2312");
                BufferedReader bufr = new BufferedReader(isr);
                String line = null;             //读取服务端传出数据
                while((line = bufr.readLine()) != null){
                    System.out.println(line);
                    osw.write(line);
                    osw.write("\r\n");
                    osw.flush();
                }
                fos.close();
                break;
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }    //打印服务端数据
    }
}
```

b) Fun2Frame.java:

```
public class Fun2Frame extends JFrame implements ActionListener{
    private Socket s;

    private JButton fun2Button = new JButton("生成柱状图");
    private JButton mainButton = new JButton("返回");
    private JPanel  fun2Panel  = new JPanel();

    public Fun2Frame(Socket s) {
        this.s = s;

        //绑定
        fun2Button.addActionListener(this);
        mainButton.addActionListener(this);
        fun2Panel.add(fun2Button);
        fun2Panel.add(mainButton);
```

```
        this.setTitle("功能二");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.add(fun2Panel,BorderLayout.NORTH);
        this.add(new JLabel(new
ImageIcon("C:\\Users\\87776\\Desktop\\images.jpg")));
        this.setSize(450,300);
        this.setVisible(true);
        this.setLocationRelativeTo(null);
    }

    public void actionPerformed(ActionEvent e) {

        if(e.getSource() == fun2Button) {
            try {
                OutputStream out = s.getOutputStream();           //获取服务端
的输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);

                bufw.println(22);                                 //发送数据给服
务端
                this.setTitle("正在下载柱状图...");
                this.setTitle("柱状图下载完成");
                bufw.println(0);
            } catch (Exception e2) {
                System.out.println(e2.getMessage());
            }
        }else if (e.getSource() == mainButton) {
            //返回主页面
            try {
                OutputStream out = s.getOutputStream();           //获取服务端的
输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);
                bufw.println(0);                                  //发送数据给服
务端
                System.out.println("功能二已返回\n");
                new MainFrame(s);
                this.dispose();
            } catch (Exception e2) {
                System.out.println(e2.getMessage());
            }
        }
    }
}
```

c) Fun3Frame.java:

```
public class Fun3Frame extends JFrame implements ActionListener, FocusListener{

    private Socket s;
    private Map<Integer, MainName> map;

    private JButton fun3Button = new JButton("保存密度图");
    private JButton mainButton = new JButton("返回");
    private JPanel  fun3Panel  = new JPanel();
    private JTextField tfMsg   = new JTextField();
    private JTextArea  taMsg   = new JTextArea("以下是十个人名对应的序号\n");

    public Fun3Frame(Socket s, Map<Integer, MainName> map) {
        this.s = s;
        this.map = map;

        Font font = new Font("黑体", 0, 20);
        taMsg.setFont(font);

        this.setLayout(new FlowLayout());

        tfMsg.setText("请输入你想查询人的序号");
        tfMsg.setForeground(Color.GRAY);

        //绑定
        fun3Button.addActionListener(this);
        mainButton.addActionListener(this);
        tfMsg.addFocusListener(this);
        fun3Panel.add(fun3Button);
        fun3Panel.add(mainButton);

        this.setTitle("功能三");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.add(fun3Panel,BorderLayout.NORTH);
        this.add(tfMsg);
        this.add(taMsg);
        this.setSize(800,325);
        this.setVisible(true);
        this.setLocationRelativeTo(null);

        for (int i = 0; i < 10; i++) {
            taMsg.append((i + 1) + ":" + map.get(i).getName() + "\n");
        }
```

```
    }

    public void actionPerformed(ActionEvent e) {
        if(e.getSource() == fun3Button) {
            try {
                OutputStream out = s.getOutputStream();
//获取服务端的输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);
                if(tfMsg.getText() == null) {
                    taMsg.append("请输入你想查询人的序号");
                }else if (tfMsg.getText().length() > 1) {
                    bufw.println(40);
                    System.out.println();
                } else {
                    bufw.println(3 + tfMsg.getText());
//发送数据给服务端
                }
            } catch (Exception e2) {
                System.out.println(e2.getMessage());
            }
        }else {
            //返回主页面
            try {
                OutputStream out = s.getOutputStream();
//获取服务端的输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);
                bufw.println(0);
//发送数据给服务端
                new MainFrame(s);
                this.dispose();
            } catch (Exception e2) {
                System.out.println(e2.getMessage());
            }
        }
    }

    public void focusGained(FocusEvent e) {
        //当获得焦点时
        String temp = tfMsg.getText();
        if(temp.equals("请输入你想查询人的序号")) {
            tfMsg.setText("");
            tfMsg.setForeground(Color.black);
        }
    }
```

```
    public void focusLost(FocusEvent e) {
        //当失去焦点时
        String temp = tfMsg.getText();
        if(temp.equals("")) {
            tfMsg.setForeground(Color.GRAY);
            tfMsg.setText("请输入你想查询人的序号");
        }
    }
}
```

d) Fun4Frame.java:

```
public class Fun4Frame extends JFrame implements ActionListener{
    private Socket s = null;

    private JButton fun1Button = new JButton("保存归类内容");
    private JButton mainButton = new JButton("返回");
    private JPanel  fun4Panel  = new JPanel();

    public Fun4Frame(Socket s) {
        this.s = s;

        //绑定
        fun1Button.addActionListener(this);
        mainButton.addActionListener(this);
        fun4Panel.add(fun1Button);
        fun4Panel.add(mainButton);

        this.setTitle("功能四");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.add(fun4Panel,BorderLayout.NORTH);
        this.setSize(800,500);
        this.setVisible(true);
        this.setLocationRelativeTo(null);

    }
    public void actionPerformed(ActionEvent e) {
        if(e.getSource() == fun1Button) {
            try {
                OutputStream out = s.getOutputStream();          //获取服务端
的输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);
                bufw.println(44);
            } catch (Exception e2) {
```

```
                System.out.println(e2.getMessage());
            }
        }else {
            //返回主页面
            try {
                OutputStream out = s.getOutputStream();          //获取服务端的
输出流，为了向服务端输出数据
                PrintWriter bufw = new PrintWriter(out,true);
                bufw.println(0);                                 //发送数据给服
务端
                this.dispose();
                new MainFrame(s);
            } catch (Exception e2) {
                System.out.println(e2.getMessage());
            }
        }
    }
}
```

e) MainFrame.java:

```
public class MainFrame extends JFrame implements ActionListener{
    private Socket s;

    private JButton fun1Button = new JButton("功能1");
    private JButton fun2Button = new JButton("功能2");
    private JButton fun3Button = new JButton("功能3");
    private JButton fun4Button = new JButton("功能4");

    public MainFrame(Socket s) {
        this.s = s;

        //四个button进行绑定
        fun1Button.addActionListener(this);
        fun2Button.addActionListener(this);
        fun3Button.addActionListener(this);
        fun4Button.addActionListener(this);

        this.setLayout(new FlowLayout());

        this.add(fun1Button);
        this.add(fun2Button);
        this.add(fun3Button);
        this.add(fun4Button);
```

```
        this.add(new JLabel(new ImageIcon("C:\\Users\\87776\\Desktop\\images
(1).jpg")));

        this.setTitle("功能选择界面");
        this.setSize(450,250);
        this.setVisible(true);
        this.setLocationRelativeTo(null);
    }

    public void actionPerformed(ActionEvent e) {
        OutputStream out = null;
        PrintWriter bufw = null;
        try {
            out = s.getOutputStream();          //获取服务端的输出流，为了向服务
端输出数据
            bufw = new PrintWriter(out,true);
        } catch (Exception e1) {
             System.out.println(e1.getMessage());
        }

        if(e.getSource() == fun1Button) {
            //显示功能一
            bufw.println(1);                                    //发送数据给服务端
            this.dispose();
            new Fun1Frame(s);
        }
        else if (e.getSource() == fun2Button) {
            //显示功能二
            bufw.println(2);                                    //发送数据给服务端
            this.dispose();
            new Fun2Frame(s);
        }
        else if (e.getSource() == fun3Button) {
            //显示功能三
            bufw.println(3);                                    //发送数据给服务端
            this.dispose();
            try {
                new Fun3Frame(s, new Name().getMap());
            } catch (IOException e1) {
                System.out.println(e1.getMessage());
            }
        }
        else {
            //显示功能四
```

```
            bufw.println(4);                                    //发送数据给服务端
            this.dispose();
            new Fun4Frame(s);
        }
    }
}
```

4. name 包

a) Name.java:

```
public class Name {

    public class MainName {

        private String name;
        private int[] num = new int[17];    //每章出现的次数
        private double avgNum;              //出现平均的章数
        private int sumNum;                 //总共出现次数

        public MainName(String name) {
            this.name = name;
        }

        public void setNum(int num, int flag) {
            this.num[flag] = num;
        }

        public int[] getNum() {
            return num;
        }

        public String getName() {
            return name;
        }

        public void setSumNum() {
            for(int flag : num) {
                sumNum += flag;
            }

        }

        public int getSumNum() {
            return sumNum;
```

```
    }

    public void setAvgNum() {
        for (int i = 0; i < num.length; i++) {
            avgNum += (((double)num[i] / (double)sumNum) * (double)(i + 1));
        }
        avgNum /= 17;
        avgNum *= 10;
    }

    public double getAvgNum() {
        return this.avgNum;
    }
}

private String[] name = {
        "哈利",
        "罗恩",
        "赫敏",
        "邓布利多",
        "海格",
        "纳威",
        "马尔福",
        "斯内普",
        "德思礼",
        "奇洛"
};

private void updateName(Map<Integer, MainName> map) {
    for (int i = 0; i < map.size(); i++) {
        for (int j = 0; j < map.size() - 1 - i; j++) {
            if(map.get(j).getSumNum() > map.get(j + 1).getSumNum()) {
                MainName temp = map.get(j);
                map.put(j, map.get(j + 1));
                map.put((j + 1), temp);
            }
        }
    }
}

private Map<Integer, MainName> map = new HashMap<Integer,MainName>();

private void setMap(String[] name) {
```

```
        int flag = 0;

        for(String mainName : name) {
            map.put(flag, (new MainName(mainName)));
            flag++;
        }

    }

    public void print(int flag) {
        System.out.println("名字: " + map.get(flag).getName() + "\n");
        for (int j = 0; j < 17; j++) {
            System.out.println("第" + (j + 1) + "章里，他出现了" +
map.get(flag).getNum()[j] + "次\n");
        }

        getPieChart(flag);
    }

    public void printAvgNum() {
        for (int i = 0; i < map.size(); i++) {
            map.get(i).setAvgNum();
            System.out.println(
                    map.get(i).getName() + "出现的相对位置是:" +
(map.get(i).getAvgNum()));
        }
    }

    public void printSum() {
        int sum =0;
        for (int i = 0; i < 10; i++) {
            sum = map.get(i).getSumNum();
            System.out.println(map.get(i).getName() + "一共出现了" + sum + "
次\n");
        }
    }

    public void getBarChart() {
        BarChart bc = new BarChart(map);

        JDialog jd=new JDialog();
        jd.setBounds(50, 50, 1400, 1200);
        jd.add(bc.getPanel());
        jd.setVisible(true);
```

```
}

public void getBarChart(int i) {
    BarChart bc = new BarChart(map, 0);

    JDialog jd=new JDialog();
    jd.setBounds(50, 50, 1400, 1200);
    jd.add(bc.getPanel());
    jd.setVisible(true);
}

public void getPieChart(int i) {
    PieChart pc = new PieChart(map.get(i));

    JDialog jd=new JDialog();
    jd.setBounds(50, 50, 1400, 1000);
    jd.add(pc.getPanel());
    jd.setVisible(true);
}

public Map<Integer, MainName> getMap() {
    return map;
}

public String[] getName() {
    return name;
}

public Name() throws IOException {
    setMap(name);

    File file = new File("C:\\Users\\87776\\Desktop\\HarryPotter\\test");
    File[] fl = file.listFiles();

    for(int flag = 0; flag < 10; flag ++) {      //flag为第几个人
        int flagNum = 0;        //flagNum为第几章的一个标志
        for (File ft : fl) {
            FileInputStream fis = new FileInputStream(ft);
            InputStreamReader isr = new InputStreamReader(fis, "utf-8");
            BufferedReader br = new BufferedReader(isr);
            StringBuffer c = new StringBuffer();
            String str = null;
            int i = 0;
            while ((str = br.readLine()) != null) {
```

```
                    c.append(str);
                }
                Pattern pattern = Pattern.compile(map.get(flag).getName());
                Matcher mather = pattern.matcher(c);
                while(mather.find()) {
                    i++;
                }
                map.get(flag).setNum(i, flagNum);
                fis.close();
                flagNum++;
            }
        }

        for (int i = 0; i < 10; i++) {
            map.get(i).setSumNum();
        }

        updateName(map);
    }

}
```

b) BarChart.java:

```
public class BarChart {
    ChartPanel jframe;

    private List<MainName> list1 = new ArrayList<MainName>();    //第一类
    private List<MainName> list2 = new ArrayList<MainName>();    //第二类
    private List<MainName> list3 = new ArrayList<MainName>();    //第三类

    public BarChart(Map<Integer, MainName> map, int x) {
        // TODO 自动生成的构造函数存根
        JTextArea taMsg = new JTextArea("分类结果:\n");
        setList(map);

        DefaultCategoryDataset data = (DefaultCategoryDataset)
getDataSet2(map);
        JFreeChart chart = ChartFactory.createBarChart3D(
                "出现次数",
                "章节",
                "次数",
                data,
                PlotOrientation.VERTICAL,
                true, false, false);
```

```
        CategoryPlot plot = chart.getCategoryPlot();
    //获得图表区域对象
        CategoryAxis domain = plot.getDomainAxis();
//水平底部列表
        domain.setTickLabelFont(new Font("黑体", Font.BOLD, 10));
//垂直标题字体设置
        domain.setLabelFont(new Font("黑体", Font.BOLD, 10));
//水平底部标题设置
        ValueAxis rangeAxis = plot.getRangeAxis();
//获取柱状体
        rangeAxis.setLabelFont(new Font("黑体", Font.BOLD, 10));
        chart.getLegend().setItemFont(new Font("黑体", Font.BOLD, 16));
//设置legend字体
        chart.getTitle().setFont(new Font("黑体", Font.BOLD, 16));

        jframe = new ChartPanel(chart);
        jframe.add(taMsg);
        jframe.setLayout(new VerticalLayout());
        Font font = new Font("黑体", 0, 20);     //设置字体格式
        taMsg.setFont(font);

        taMsg.append("第一类: ");
        for (int i = 0; i < list1.size(); i++) {
            taMsg.append(list1.get(i).getName() + " ");
        }
        taMsg.append("\n第二类: ");
        for (int i = 0; i < list2.size(); i++) {
            taMsg.append(list2.get(i).getName() + " ");
        }
        taMsg.append("\n第三类: ");
        for (int i = 0; i < list3.size(); i++) {
            taMsg.append(list3.get(i).getName() + " ");
        }

        try {
            FileOutputStream fos = new FileOutputStream(

    "C:\\Users\\87776\\Desktop\\HarryPotter\\Fun4BarChart.jpg");
            ChartUtilities.writeChartAsJPEG(fos, chart, 1000, 700);
            fos.close();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
```

```
    }

    public BarChart(Map<Integer, MainName> map) {
        JTextArea taMsg = new JTextArea("存在感排名为：\n");

        DefaultCategoryDataset data = (DefaultCategoryDataset)
getDataSet1(map);
        JFreeChart chart = ChartFactory.createBarChart3D(
                "人名出现次数",
                "人名",
                "次数",
                data,
                PlotOrientation.VERTICAL,
                true, false, false);

        CategoryPlot plot = chart.getCategoryPlot();
    //获得图表区域对象
        CategoryAxis domain = plot.getDomainAxis();
//水平底部列表
        domain.setTickLabelFont(new Font("黑体", Font.BOLD, 10));
//垂直标题字体设置
        domain.setLabelFont(new Font("黑体", Font.BOLD, 10));
//水平底部标题设置
        ValueAxis rangeAxis = plot.getRangeAxis();
//获取柱状体
        rangeAxis.setLabelFont(new Font("黑体", Font.BOLD, 10));
        chart.getLegend().setItemFont(new Font("黑体", Font.BOLD, 16));
//设置legend字体
        chart.getTitle().setFont(new Font("黑体", Font.BOLD, 16));

        jframe = new ChartPanel(chart);
        jframe.add(taMsg);
        jframe.setLayout(new VerticalLayout());
        Font font = new Font("黑体", 0, 20);     //设置字体格式
        taMsg.setFont(font);                    //更改taMsg字体格式

        for (int i = 0; i < map.size(); i++) {
            taMsg.append(map.get(i).getName() + " ");

        }

        try {
            FileOutputStream fos = new FileOutputStream(
                    "C:\\Users\\87776\\Desktop\\HarryPotter\\BarChart.jpg");
```

```
            ChartUtilities.writeChartAsJPEG(fos, chart, 1000, 700);
            fos.close();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }

    public ChartPanel getPanel() {
        return jframe;
    }

    private void setList(Map<Integer, MainName> map) {
        for (int i = 0; i < map.size(); i++) {
            double avgNum = map.get(i).getAvgNum();
            if(avgNum <= 3) {
                list1.add(map.get(i));
            }else if (avgNum >= 7.5) {
                list3.add(map.get(i));
            }else {
                list2.add(map.get(i));
            }
        }
    }

    private CategoryDataset getDataSet1(Map<Integer, MainName> map) {
         DefaultCategoryDataset data=new DefaultCategoryDataset();
         //设置数据
         for (int i = 0; i < 10; i++) {
            data.setValue(map.get(i).getSumNum(), "", map.get(i).getName());
        }
         return data;
    }

    private CategoryDataset getDataSet2(Map<Integer, MainName> map) {
         DefaultCategoryDataset data=new DefaultCategoryDataset();
         //设置数据
         for (int i = 0; i < 17; i++) {
            for (int j = 0; j < map.size(); j++) {
                data.setValue(map.get(j).getAvgNum(), "",
map.get(j).getName());
            }
        }
         return data;
    }
```

```
}
```

c) PieChart.java:

```
public class PieChart {
    ChartPanel jframe;

    public PieChart(MainName mn) {
        DefaultPieDataset data = (DefaultPieDataset) getPieDataset(mn);
        JFreeChart chart = ChartFactory.createPieChart3D(
                mn.getName() + "各章节出现密度", data,
                true, false, false);
        chart.setAntiAlias(false);   //关闭锯齿形

        Font font = new Font("黑体", Font.PLAIN, 14);

        PiePlot plot = (PiePlot)chart.getPlot();
        //解决图像不能显示中文
        plot.setLabelFont(font);
        LegendTitle lt = chart.getLegend();
        lt.setItemFont(font);

        //按照出现频率大小来设置颜色深浅
        setColor(plot, data, mn);

        //设置字体
        TextTitle chartTitle = chart.getTitle();
        chartTitle.setFont(new Font(
                "黑体", Font.BOLD, 20));

        //设置百分比
        plot.setLabelGenerator(new StandardPieSectionLabelGenerator("{1}次
"));

        jframe = new ChartPanel(chart);

        //下载饼状图
        try {
            FileOutputStream fos = new FileOutputStream(
                    "C:\\Users\\87776\\Desktop\\HarryPotter\\PieChart.jpg");
            ChartUtilities.writeChartAsJPEG(fos, chart, 1000, 700);
            fos.close();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
```

```
    }

    public ChartPanel getPanel() {
        return jframe;
    }

    private void setColor(PiePlot plot, PieDataset data, MainName mn) {
        List keys = data.getKeys();
        for (int i = 0; i < keys.size(); i++) {
            plot.setSectionPaint(keys.get(i).toString(),new Color(
                    0,  mn.getNum()[i] * 1000 / mn.getSumNum() / 4,
mn.getNum()[i] * 1000 / mn.getSumNum() / 4));
        }
    }

    private PieDataset getPieDataset(MainName mn){
        DefaultPieDataset dataset = new DefaultPieDataset();
        int[] num = mn.getNum();

        for (int i = 0; i < num.length; i++) {
            dataset.setValue("第" + (i + 1) + "章", num[i]);
        }
        return dataset;
    }
}
```

实验总结

通过这次课程设计，大大的提高了我对 Java 语言的认识，更加熟悉了 Java 语言的编程过程以及编程风格，提高了我自身的能力。

在完成此次课设的时候，我也遇到了不少的问题，但已经全部通过自己在网上查找资料解决。我记忆最深的就是当我点击完第一个功能的 button 之后再点击其他 button 就会没有反应，这个 bug 困扰了我好久，最后我才在网上找到原因，是因为我 InputStream is = s.getInputStream();写完这句话的时候随手写了 is.close();而导致了整个 s 的关闭。还有在最开始的编写过程中，在写 try 和 catch 模块时，我总是不会在 catch 里写东西，而导致我在之后的 debug 时找不到问题所在。

在通过这次出现错误，又不断更正错误的过程中，我也更加熟悉了 Java 的 debug 功能使用，更加的熟练的运用控制台来帮助自己找出程序中的错误，两个星期的编程中，我提升了很多。