

Java 程序设计实验报告

学号： 1173710105

姓名： 曾钰城

专业： 软件工程

班级： 1737101

哈尔滨工业大学

实验六：文件读写程序设计与异常

一、实验目的

- 1) 了解流输入输出的基本概念。
- 2) 了解异常的概念。
- 3) 掌握 File、FileReader、FileWriter 等文件读写类库的使用。
- 3) 掌握 IOException、FileNotFoundException 等异常的使用。
- 4) 掌握 Swing 常用组件及绝对布局的应用
- 5) 掌握事件驱动编程原理及应用
- 6) 掌握菜单组件的设计与使用。
- 7) 掌握对话框(Dialog)的使用。

二、实验内容

- 1) 将 CollectionBMI 类改造为 GUIFileBMI 类;
- 2) 在 GUIFileBMI 类中增加方法 saveFile(ArrayList<Student> students, String filename), 该方法可以将所有学生信息以逗号表的形式写入到当前目录下 XXX.txt 文件中(XXX 为您的学号), 每一行写入一个学生。
- 3) 在 GUIFileBMI 类中增加方法 ArrayList<Student> readFile(String filename)读文件中的数据到学生 students 中。
- 4) 在 GUIFileBMI 类中增加 4 个界面(Panel): 学生信息主界面、学生信息输入界面、学生信息维护界面、BMI 统计界面;
 - a) 学生信息主界面为默认界面, 该界面可从文件中读取并显示所有学生信息, 并按不同属性对学生排序显示。该界面还可随机生成 200 名学生, 并保存到文件中。
 - b) 学生信息输入界面每次输入单个学生, 保存后, 提示用户是否成功保存或提示用户已经存在, 请重新输入(利用 Dialog 对话框进行提示)。
 - c) 学生信息维护界面要求输入要维护的学生学号, 按学号查询并显示该学生所有信息, 可删除或修改学生信息, 并提示用户修改/删除成功或失败 (利用 Dialog 对话框进行提示)。
 - d) BMI 统计界面显示 bmi 的相关统计信息 (均值、中值、众数、方差等), 并将 bmi 值的范围由小到大划分的 10 个均分区间, 统计每个区间的学生人数, 并绘制出相应的柱状图(利用 JFreechart 实现)。
- 5) 在 GUIFileBMI 类中增加**菜单**, 可通过菜单切换显示上述 4 各不同界面。
注意, 身高、体重、及 bmi 等数值均需保留两位小数的格式进行存储和显示。

三、实验步骤

注意：将程序代码和运行结果截图粘贴在此处，注意源代码中注释行数不少于全部代码的1/3，程序源代码请压缩后上传，压缩文件按照 学号.zip 进行命名，注意源程序与报告请分别上传到不同的文件夹中！

程序代码：

```
package edu.hit.java.exp4.hit1173710105;

import java.awt.*;
import javax.swing.*;
import javax.swing.border.EmptyBorder;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.axis.CategoryAxis;
import org.jfree.chart.axis.ValueAxis;
import org.jfree.chart.plot.CategoryPlot;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.chart.renderer.category.BarRenderer3D;
import org.jfree.data.category.CategoryDataset;
import org.jfree.data.category.DefaultCategoryDataset;
import java.awt.Font;
import java.awt.event.*;
import java.io.*;
import java.util.*;

public class GUIFileBMI extends JFrame
{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private JPanel contentPane;

    /**
     * Launch the application.
     */
    public static void main(String[] args)
    {
        EventQueue.invokeLater(new Runnable()
        {
```

```

        public void run()
        {
            try
            {
                GUIFileBMI frame = new GUIFileBMI();
                frame.setVisible(true);
            } catch (Exception e)
            {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public GUIFileBMI()
{
    getContentPane().setLayout(null); //使用绝对布局
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); //使用 System exit 方法退出
应用程序
    setBounds(100, 100, 700, 600); //设置窗体大小

    JMenuBar menuBar = new JMenuBar(); //增加菜单栏
    setJMenuBar(menuBar);

    JMenu HelpMenu = new JMenu("Help"); //增加 Help 菜单
    HelpMenu.setFont(new Font("Microsoft YaHei UI", Font.PLAIN, 20));
    menuBar.add(HelpMenu);
    JMenuItem About = new JMenuItem("About"); //Help 菜单中增加 About 选项
    About.setFont(new Font("Microsoft YaHei UI", Font.PLAIN, 18));
    HelpMenu.add(About);
    About.addActionListener(new ActionListener() //点击 About 后弹窗
    {
        @Override
        public void actionPerformed(ActionEvent e)
        {
            JOptionPane.showMessageDialog(null, "这是一个学生信息管理系统",
"About", JOptionPane.INFORMATION_MESSAGE);
        }
    });

    JMenu mnNewMenu = new JMenu("Fcuntion"); //增加 Fcuntion 菜单

```

```
mnNewMenu.setFont(new Font("Microsoft YaHei UI", Font.PLAIN, 20));
menuBar.add(mnNewMenu);
```

```
JMenuItem AllInfo = new JMenuItem("学生信息总览");//增加学生信息总览选项
AllInfo.setFont(new Font("Microsoft YaHei UI", Font.PLAIN, 18));
mnNewMenu.add(AllInfo);
AllInfo.addActionListener(new ActionListener()//点击后进入总览学生信息界面
{
```

```
    @Override
    public void actionPerformed(ActionEvent e)
    {
        showStudetns();
        setSize(700, 950);
    }
});
```

选项

```
JMenuItem ModifyStudent = new JMenuItem("修改学生信息");//增加学生信息总览
```

```
ModifyStudent.setFont(new Font("Microsoft YaHei UI", Font.PLAIN, 18));
mnNewMenu.add(ModifyStudent);
```

面

```
ModifyStudent.addActionListener(new ActionListener()//点击后进入修改学生信息界
```

```
{
    @Override
    public void actionPerformed(ActionEvent e)
    {
        showModify();
        setSize(700, 600);
    }
});
```

项

```
JMenuItem AddStudent = new JMenuItem("增加学生信息");//增加 增加学生信息选
```

```
mnNewMenu.add(AddStudent);
AddStudent.setFont(new Font("Microsoft YaHei UI", Font.PLAIN, 18));
AddStudent.addActionListener(new ActionListener()//点击之后进入增加学生界面
{
```

```
    @Override
    public void actionPerformed(ActionEvent e)
    {
        showAddStudent();
        setSize(700, 600);
    }
});
```

```

    }
});

JMenuItem BMISatic = new JMenuItem("BMI 统计信息");//增加 BMI 统计信息选项
mnNewMenu.add(BMISatic);
BMISatic.setFont(new Font("Microsoft YaHei UI", Font.PLAIN, 18));
BMISatic.addActionListener(new ActionListener()//点击之后进入 BMI 统计信息界面
{

    @Override
    public void actionPerformed(ActionEvent e)
    {
        showBMISatic();
        setSize(1650,1080);
    }
});

contentPane = new JPanel();
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
contentPane.setLayout(new BorderLayout(0, 0));
setContentPane(contentPane);

}

private void showStudetns()
{
    this.setContentPane(new resultPanel());
    setVisible(true);
}

private void showModify()
{
    this.setContentPane(new ModifyPanel());
    setVisible(true);
}

private void showAddStudent()
{
    this.setContentPane(new addStudentPanel());
    setVisible(true);
}

private void showBMISatic()
{

```

```

        this.setContentPane(new BMIStaticPanel());
        setVisible(true);
    }
}

class BMIStaticPanel extends JPanel
{
    /**
     *
     */
    private static final long serialVersionUID = 1L;

    public BMIStaticPanel()
    {
        ArrayList<Student> studentlist = new ArrayList<Student>();
        HandleFileTool.readFile(studentlist, "StudentInfo.txt");//从文件之中读取信息，然后把
        信息放到数组列表 studentlist

        JLabel AverageLabel = new JLabel("平均值:"+calculateAverageOfBMI(studentlist));//
        显示平均值
        AverageLabel.setFont(new Font("微软雅黑", Font.PLAIN, 20));// 设置标题字体
        AverageLabel.setBounds(300, 30, 150, 30);//设置位置

        JLabel MedianLabel = new JLabel("中位数:"+calculateMedian(studentlist));//显示中位
        数
        MedianLabel.setFont(new Font("微软雅黑", Font.PLAIN, 20));// 设置标题字体
        MedianLabel.setBounds(600, 30, 150, 30);//设置位置

        JLabel ModeLabel = new JLabel("众数:"+findMode(studentlist));//显示众数
        ModeLabel.setFont(new Font("微软雅黑", Font.PLAIN, 20));// 设置标题字体
        ModeLabel.setBounds(800, 30, 150, 30);//设置位置

        JLabel VarianceLabel = new JLabel("方差:"+calculateVariance(studentlist));//显示方差
        VarianceLabel.setFont(new Font("微软雅黑", Font.PLAIN, 20));// 设置标题字体
        VarianceLabel.setBounds(1100, 30, 150, 30);//设置位置

        this.setLayout(null);//绝对布局
        CategoryDataset dataset = getDataSet(studentlist);
        JFreeChart chart = ChartFactory.createBarChart3D("学生 BMI 值统计数据", // 图表标
        题
            "BMI 值", // 目录轴的显示标签
            "学生人数", // 数值轴的显示标签
            dataset, // 数据集
            PlotOrientation.VERTICAL, // 图表方向：水平、垂直

```

```

        true, // 是否显示图例(对于简单的柱状图必须是 false)
        true, // 是否生成工具
        false // 是否生成 URL 链接
    );
    // 从这里开始
    CategoryPlot plot = chart.getCategoryPlot();// 获取图表区域对象
    CategoryAxis domainAxis = plot.getDomainAxis(); // 水平底部列表
    domainAxis.setLabelFont(new Font("黑体", Font.PLAIN, 14)); // 水平底部标题
    domainAxis.setTickLabelFont(new Font("宋体", Font.PLAIN, 12)); // 垂直标
    ValueAxis rangeAxis = plot.getRangeAxis();// 获取柱状
    BarRenderer3D renderer = new BarRenderer3D();
    renderer.setItemMargin(0.0000001);
    renderer.setMaximumBarWidth(0.05);//设置柱形宽度
    rangeAxis.setLabelFont(new Font("微软雅黑", Font.PLAIN, 15));
    chart.getLegend().setItemFont(new Font("黑体", Font.PLAIN, 15));
    chart.getTitle().setFont(new Font("宋体", Font.PLAIN, 20));// 设置标题字体
    ChartPanel frame1=new ChartPanel(chart,true);          //这里也可以用 chartFrame,
    可以直接生成一个独立的 Frame
    frame1.setBounds(0,100,1600,830);
    plot.setRenderer(renderer);

    this.add(AverageLabel);
    this.add(MedianLabel);
    this.add(ModeLabel);
    this.add(VarianceLabel);
    this.add(frame1);
}

/**
 * 设置数据集，返回 dataset，用书柱形图的数据*/
private CategoryDataset getDataSet(ArrayList<Student> studentlist)
{
    DefaultCategoryDataset dataset = new DefaultCategoryDataset();
    int[] range = calculateRange(studentlist);//获取每个 BMI 值范围内有多少人
    dataset.addValue(range[0], "10.5~13.5", "10.5~13.5");
    dataset.addValue(range[1], "13.5~17.0", "13.5~17.0");
    dataset.addValue(range[2], "17.0~20.5", "17.0~20.5");
    dataset.addValue(range[3], "20.5~24.0", "20.5~24.0");
    dataset.addValue(range[4], "24.0~27.5", "24.0~27.5");
    dataset.addValue(range[5], "27.5~31.0", "27.5~31.0");
    dataset.addValue(range[6], "31.0~34.5", "31.0~34.5");
    dataset.addValue(range[7], "34.5~38.0", "34.5~38.0");
    dataset.addValue(range[8], "38.0~41.5", "38.0~41.5");
    dataset.addValue(range[9], "41.5~45.0", "41.5~45.0");
}

```



```

        return dataset;
    }

    /**获取每个 BMI 值范围内有多少人,返回一个一位数组*/
    private int[] calculateRange(ArrayList<Student> studentlist)
    {
        int[] range = new int[10];
        for(int i=0;i<10;i++)
            range[i] = 0;
        Float BMI;
        for(int i=0;i<studentlist.size();i++)
        {
            BMI = studentlist.get(i).getBMI();
            if(10.5<=BMI&&BMI<13.5)
                range[0]++;
            else if(13.5<=BMI&&BMI<17)
                range[1]++;
            else if(17<=BMI&BMI<20.5)
                range[2]++;
            else if(20.5<=BMI&BMI<24)
                range[3]++;
            else if(24<=BMI&BMI<27.5)
                range[4]++;
            else if(27.5<=BMI&BMI<31)
                range[5]++;
            else if(31<=BMI&BMI<34.5)
                range[6]++;
            else if(34.5<=BMI&BMI<38)
                range[7]++;
            else if(38<=BMI&BMI<41.5)
                range[8]++;
            else if(41.5<=BMI&BMI<=45)
                range[9]++;
            else {
            }
        }
        return range;
    }

    /** 计算所有学生的 BMI 的平均值 */
    private float calculateAverageOfBMI(ArrayList<Student> studentlist)
    {
        float average = 0;
        for (Student result : studentlist)//用 for—loop 循环，从数组列表 studentlist 里面获取
        每一个实例化对象

```

```

        average += result.getBMI();//获取每一个实例化对象对 BMI 值
return (Math.round( average / studentlist.size() * 100)) / 100;
}

/**
 * 计算所有学生的 BMI 的中位数
 */
private float calculateMedian(ArrayList<Student> studentlist)
{
    Student student = new Student();//创建一个新对象，用与创建比较器对象
    //调用 Collection.sort(), 传进一个重写的比较器，实现对 BMI 升序排序
    Collections.sort(studentlist, student.new SortByBMIAsc());
    int lenght = studentlist.size();//记录学生人数
    float Median;
    if (lenght % 2 != 0)// 如果 bmi 数组长度是奇数，即有奇数个学生，则中位数就是数
    组索引值为 lenght/2 的值
    {
        Median = studentlist.get(lenght / 2).getBMI();
    } else // 如果 bmi 数组长度是偶数，即有偶数个学生，则中位数就是数组索引值为
    lenght/2 与 lenght/2+1 的值的平均值
    {
        Median = (studentlist.get(lenght / 2 + 1).getBMI() + studentlist.get(lenght /
    2).getBMI()) / 2;
    }
    return (Math.round( Median * 100)) / 100;
}

/**
 * 用于寻找 student 对象数组里面所有学生各自 BMI 值的众数
 */
private String findMode(ArrayList<Student> studentlist)
{
    Student student = new Student();//创建一个新对象，用与创建比较器对象
    //调用 Collection.sort(), 传进一个重写的比较器，实现对 BMI 升序排序
    Collections.sort(studentlist, student.new SortByBMIAsc());
    int length = studentlist.size();// 记录学生的个数，即数组 BMI 的长度
    float[] bmis = new float[length];// 用于获得 student 对象数组列表里面每个学生的
    BMI 值
    for (int i = 0; i < length; i++)
    {
        bmis[i] = studentlist.get(i).getBMI();//获取象数组列表里面每个学生的 BMI 值
    }
    float[] num = new float[length];// 用于存储记录不同得 BMIS 值

```

```

int[] count = new int[length];// 用于记录不同的 BMI 在数组中出现的次数
int time = 0;// 用于计数，没出现一个相同的 BMI 值，自增一次
int k = 0;// 用于 num 与 count 数组下标的自增

for (int i = 0; i < length; i++)
{
    if (bmis[i] > 0)// 判断 bims[i]里面存放的是否是有用数据
    {
        num[k] = bmis[i];// 记录这个数值
        for (int j = 0; j < length; j++)
        {
            if (bmis[j] > 0 && bmis[j] == num[k])// 判断 bims[j]里面存放的是否
            是有用数据且判断是否与 num[k]相等
            {
                time++;// 这个数出现次数加一
                bmis[j] = 0f;// 将这个数置为无用数据，以后不再参与统计
            }
        }
        count[k] = time;// 记录出现次数
        k++;
        time = 0;// 重置计数器
    }
}

float max = count[0];
// 找到出现次数最多的 BMI 值,若果 max = 1,
// BMI 中所有数均只出现一次，每个数都不一样，那么众数不存在
for (int i = 0; i < count.length; i++)
{
    if (count[i] > max)
    {
        max = count[i];
    }
}

// 找到出现次数最多的数据的下标，并记录下来
int[] ModeSub = new int[length];
k = 0;
for (int i = 0; i < length; i++)
{
    if (count[i] == max)
    {
        ModeSub[k] = i;
        k++;
    }
}

```

```

    }
}

if (max == 1)// BMI 中所有数均只出现一次，每个数都不一样，那么众数不存在
{
    //System.out.println("所有学生的 BMI 值均只出现一次，众数不存在！");
    return "无众数";
}

//System.out.print("\n" + "众数为:");
String mode = "";
for (int i = 0; i < length; i++)
{
    if (ModeSub[i] != 0)// 过滤非法数据
    {
        mode = (Math.round( num[ModeSub[i]] * 100)) / 100 + ",";
    }
}
return mode.substring(0, mode.length()-1);
}

/**
 * 计算所有学生的 BMI 的方差
 */
private float calculateVariance(ArrayList<Student> studentlist)
{
    float average = 0;// 计算所有学生的 BMI 平均值
    float sum = 0;// 计算总值
    int length = studentlist.size();// 记录学生的个数，即数组 BMI 的长度
    float[] bmis = new float[length];//用于获得 student 对象数组列表里面每个学生的
BMI 值
    for (int i = 0; i < length; i++)
    {
        bmis[i] = studentlist.get(i).getBMI();//获取象数组列表里面每个学生的 BMI 值
        average = bmis[i];//计算每个学生的 BMI 值总和
    }
    average = average / length;//计算出平均值
    for (int i = 0; i < length; i++)
    {
        sum += (bmis[i] - average) * (bmis[i] - average);// 迭代计算出方差
    }
    return (Math.round( sum / length * 100)) / 100;
}

```

```

}

class addStudentPanel extends JPanel
{
    private static final long serialVersionUID = 1L;

    public addStudentPanel()
    {
        ArrayList<Student> studentlist = new ArrayList<Student>();
        HandleFileTool.readFile(studentlist, "StudentInfo.txt");//从文件之中读取信息，然后把
        信息放到数组列表 studentlist

        this.setLayout(null);
        JTextField IDjtf = new JTextField();
        IDjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        IDjtf.setBounds(120, 40, 250, 30);//设置位置

        JTextField Namejtf = new JTextField();
        Namejtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        Namejtf.setBounds(120, 90, 250, 30);//设置位置

        JTextField Heightjtf = new JTextField();
        Heightjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        Heightjtf.setBounds(120, 140, 250, 30);//设置位置

        JTextField Weightjtf = new JTextField();
        Weightjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        Weightjtf.setBounds(120, 190, 250, 30);//设置位置

        JTextField BMIjtf = new JTextField();
        BMIjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        BMIjtf.setBounds(120, 230, 250, 30);//设置位置
        BMIjtf.setEditable(false);
        JTextField PyhsicalConditionjtf = new JTextField();
        PyhsicalConditionjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        PyhsicalConditionjtf.setBounds(120, 270, 250, 30);//设置位置
        PyhsicalConditionjtf.setEditable(false);//设置为不能被选中

        JLabel IDLabel = new JLabel("    ID:");
        IDLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        IDLabel.setBounds(50, 40, 250, 30);

        JLabel NameLabel = new JLabel("姓名:");
        NameLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
    }
}

```

```

NameLabel.setBounds(50, 90, 250, 30);
JLabel HeightLabel = new JLabel("身高:");
HeightLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
HeightLabel.setBounds(50, 140, 250, 30);
JLabel WeightLabel = new JLabel("体重:");
WeightLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
WeightLabel.setBounds(50, 190, 250, 30);
JLabel BMILabel = new JLabel("BMI 值:");
BMILabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
BMILabel.setBounds(35, 230, 250, 30);
JLabel PyhsicalConditionLabel = new JLabel("健康状况:");
PyhsicalConditionLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
PyhsicalConditionLabel.setBounds(20, 270, 250, 30);

JButton comfirm = new JButton();
comfirm.setText("增加");
comfirm.setFont(new Font("微软雅黑", Font.PLAIN, 18));
comfirm.setBounds(120, 310, 100, 30);
comfirm.setEnabled(false);//在检测合法之前不能增加学生
comfirm.addActionListener(new ActionListener()
{

    @Override
    public void actionPerformed(ActionEvent e)
    {
        String id = IDjtf.getText();
        String name = Namejtf.getText();
        String height = Heightjtf.getText();
        String weight = Weightjtf.getText();
        Student student = new Student(id, name, Float.parseFloat(height),
Float.parseFloat(weight));
        studentlist.add(student);
        HandleFileTool.saveFile(studentlist, "StudentInfo.txt");//将增加的学生数组
保存文件之中
        BMIjtf.setText(student.getBMI() + "");
        PyhsicalConditionjtf.setText(student.getPhysicalCondition());
        JOptionPane.showMessageDialog(null, "增加学生信息成功", "Message",
JOptionPane.INFORMATION_MESSAGE);
        comfirm.setEnabled(false);
    }
});

JButton islegal = new JButton();
islegal.setText("合法性检测");

```

```

islegal.setFont(new Font("微软雅黑", Font.PLAIN, 18));
islegal.setBounds(250, 310, 140, 30);
islegal.addActionListener(new ActionListener()
{
    @Override
    public void actionPerformed(ActionEvent e)
    {
        String id = IDjtf.getText();
        boolean flag1 = true;
        boolean flag2 = true;
        boolean flag3 = true;
        for (int i = 0; i < studentlist.size(); i++)
        {
            //数据合法性检测, 检测学号是否是重复, 检测身高体重是否是在一
            定范围之内的浮点数
            if (id.equals(studentlist.get(i).getID()))
            {
                JOptionPane.showMessageDialog(null, "ID 已经被占用, 请重新输
                入 ID", "ERROR", JOptionPane.ERROR_MESSAGE);
                flag1 = false;
                break;
            }
        }
        if (flag1)
        {
            try
            {
                float height1 = Float.parseFloat(Heightjtf.getText());
                float weight1 = Float.parseFloat(Weightjtf.getText());
                flag2 = true;
                if (!isLegalNum(height1, 1.5f, 2.0f) || !isLegalNum(weight1, 45f,
                100f))//是否已在一定范围之内的浮点数
                {
                    JOptionPane.showMessageDialog(null, "数据不合法, 体重和
                    身高数据不合理", "Warning",
                    JOptionPane.WARNING_MESSAGE);
                    flag3 = false;
                }
            }
            catch (Exception e2)
            {
                flag2 = false;
            }
        }
    }
}

```

JOptionPane.showMessageDialog(null, "数据不合法，体重和身高
要求输入浮点数", "Warning",

JOptionPane.WARNING_MESSAGE); //不合法数据，弹
窗显示错误

```
        }  
    }  
    if (flag1 && flag2 && flag3)  
    {  
        JOptionPane.showMessageDialog(null, "数据合法", "Message",  
JOptionPane.INFORMATION_MESSAGE);  
        confirm.setEnabled(true);  
    }  
}  
});
```

```
this.add(IDjtf);  
this.add(Namejtf);  
this.add(Heightjtf);  
this.add(Weightjtf);  
this.add(IDLabel);  
this.add(NameLabel);  
this.add(HeightLabel);  
this.add(WeightLabel);  
this.add(PyhsicalConditionLabel);  
this.add(PyhsicalConditionjtf);  
this.add(BMIjtf);  
this.add(BMILabel);  
this.add(confirm);  
this.add(islegal);
```

```
}
```

```
private boolean isLegalNum(float num, float min, float max)
```

```
{  
    if (num <= max && num >= min)  
        return true;  
    else  
        return false;  
}  
}
```

```
/**
```

```
 * 学生信息总览，里面包括增加学生信息，按指定顺序输出信息的功能*/
```

```
class resultPanel extends JPanel
```



```

{
    private static final long serialVersionUID = 1L;

    public resultPanel()
    {
        this.setSize(150, 80);
        JTextArea textarea = new JTextArea("", 50, 60);
        ArrayList<Student> studentlist = new ArrayList<Student>();
        HandleFileTool.readFile(studentlist, "StudentInfo.txt");
        StringBuffer sb = new StringBuffer();
        sb.append("学号\t姓名\t身高\t体重\tBMI 值\t健康状况\n");
        //往 TextArea 里面写书数据
        for (Student st : studentlist)
        {
            sb.append(st.toString()).append("\n");
        }
        textarea.setText(sb.toString());
        textarea.setEditable(false);
        JScrollPane sp = new JScrollPane(textarea);
        sp.setBounds(5, 5, 500, 300);
        sp.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
        JButton addBtn = new JButton("增加数据");
        addBtn.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        addBtn.addActionListener(new ActionListener()
        {
            @Override
            public void actionPerformed(ActionEvent e)
            {
                String[][] data = HandleStudentInfo.genStudents(studentlist); //增加学生，将
                增加的学生信息返回到一个二维数组之中
                //将新增加的数据追加到 TextArea 末尾
                for (int i = 0; i < data.length; i++)
                {
                    for (int j = 0; j < data[i].length - 1; j++)
                    {
                        textarea.append(data[i][j] + "\t");
                    }
                    textarea.append(data[i][5] + "\n");
                }
            }
        });

        //下来列表，让用户排序方式，根据选择执行相应的排序方式
        JComboBox<String> cBox = new JComboBox<String>(

```

```

        new String[] { "请选择排序方式", "按学号从小到大排序", "按姓名从小到大排序", "按身高从小到大排序", "按体重从小到大排序", "按 BMI 值从小到大排序" });
        cBox.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        cBox.addItemListener(new ItemListener()
        {

            @Override
            public void itemStateChanged(ItemEvent e)
            {
                Student student = new Student();
                switch ((String) e.getItem())
                {
                    case "按学号从小到大排序":
                        Collections.sort(studentlist, student.new SortByIDAsc());//按学号从小到大对数组列表 studentlist 排序
                        ChangeTextArea(studentlist, textarea);//更新数组列表
                        break;
                    case "按姓名从小到大排序":
                        Collections.sort(studentlist, student.new SortByNameAsc());//按姓名从小到大对数组列表 studentlist 排序
                        ChangeTextArea(studentlist, textarea);//更新数组列表
                        break;
                    case "按身高从小到大排序":
                        Collections.sort(studentlist, student.new SortByHeightAsc());//按身高从小到大对数组列表 studentlis 排序
                        ChangeTextArea(studentlist, textarea);//更新数组列表
                        break;
                    case "按体重从小到大排序":
                        Collections.sort(studentlist, student.new SortByWeightAsc());//按体重从小到大对数组列表 studentlist 进行排序
                        ChangeTextArea(studentlist, textarea);//更新数组列表
                        break;
                    case "按 BMI 值从小到大排序":
                        Collections.sort(studentlist, student.new SortByBMIAsc());//按体重从小到大对数组列表 studentlist 进行排序
                        ChangeTextArea(studentlist, textarea);//更新数组列表
                        break;
                    default:
                        break;
                }
            }
        });
        this.add(addBtn);
        this.add(cBox);

```

```

        this.add(sp);
    }

    /**
     * 更新数组列表，将排序之后的 studentlist 重新写到 textarea 之中*/
    private void ChangeTextArea(ArrayList<Student> studentlist, JTextArea textarea)
    {
        StringBuffer sb = new StringBuffer();
        sb.append("学号\t姓名\t身高\t体重\tBMI 值\t健康状况\n");
        for (Student st : studentlist)
        {
            sb.append(st.toString()).append("\n");
        }
        textarea.setText(sb.toString());
    }
}

/**
 * 修改学生信息*/
class ModifyPanel extends JPanel
{
    private static final long serialVersionUID = 1L;
    private String ID;//固定输入学生的 ID
    private int sub;//记录这个 ID 在 studentlist 中的位置

    public ModifyPanel()
    {

        ArrayList<Student> studentlist = new ArrayList<Student>();
        HandleFileTool.readFile(studentlist, "StudentInfo.txt");//从文件之中读取信息，防止到
        塑料布 studentlist 之中

        this.setLayout(null);//绝对布局
        JTextField IDjtf = new JTextField("请输入请修改学生的学号");
        IDjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        IDjtf.setBounds(120, 40, 250, 30);

        JTextField Namejtf = new JTextField();
        Namejtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        Namejtf.setBounds(120, 90, 250, 30);
        JTextField Heightjtf = new JTextField();
        Heightjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
        Heightjtf.setBounds(120, 140, 250, 30);
    }
}

```

```

JTextField Weightjtf = new JTextField();
Weightjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
Weightjtf.setBounds(120, 190, 250, 30);

JTextField BMIjtf = new JTextField();
BMIjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
BMIjtf.setBounds(120, 230, 250, 30);
BMIjtf.setEditable(false);
JTextField PyhsicalConditionjtf = new JTextField();
PyhsicalConditionjtf.setFont(new Font("微软雅黑", Font.PLAIN, 18));
PyhsicalConditionjtf.setBounds(120, 270, 250, 30);
PyhsicalConditionjtf.setEditable(false);

JLabel IDLabel = new JLabel("ID:");
IDLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
IDLabel.setBounds(50, 40, 250, 30);
JLabel NameLabel = new JLabel("姓名:");
NameLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
NameLabel.setBounds(50, 90, 250, 30);
JLabel HeightLabel = new JLabel("身高:");
HeightLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
HeightLabel.setBounds(50, 140, 250, 30);
JLabel WeightLabel = new JLabel("体重:");
WeightLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
WeightLabel.setBounds(50, 190, 250, 30);
JLabel BMILabel = new JLabel("BMI 值:");
BMILabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
BMILabel.setBounds(35, 230, 250, 30);
JLabel PyhsicalConditionLabel = new JLabel("健康状况:");
PyhsicalConditionLabel.setFont(new Font("微软雅黑", Font.PLAIN, 18));
PyhsicalConditionLabel.setBounds(20, 270, 250, 30);

JButton check = new JButton();
check.setText("查询");
check.setFont(new Font("微软雅黑", Font.PLAIN, 18));
check.setBounds(400, 40, 100, 30);
check.addActionListener(new ActionListener()
{

    @Override
    public void actionPerformed(ActionEvent e)
    {
        String IDtext = IDjtf.getText().toString();
        boolean flag = false;

```

```

        if (IDtext != "")//检测 IDtext 里面是否为空
        {
            for (int i = 0; i < studentlist.size(); i++)
            {
                if (IDtext.equals(studentlist.get(i).getID()))
                {
                    flag = true;
                    ID = IDtext;//记录学生的 ID
                    sub = i;//记录该 ID 学生在数组列表在中的位置
                    //将该学生的信息写入到窗口里面的文本框
                    Namejtf.setText(studentlist.get(i).getName());
                    Heightjtf.setText("" + studentlist.get(i).getHeight());
                    Weightjtf.setText("" + studentlist.get(i).getWeight());
                    BMIjtf.setText("" + studentlist.get(i).getBMI());

                    PyhsicalConditionjtf.setText(studentlist.get(i).getPhysicalCondition());
                    break;
                }
            }
            if (!flag)
                JOptionPane.showMessageDialog(null, "查无此人，请重新输入学
号", "Warning", JOptionPane.WARNING_MESSAGE);
        } else
        {
            JOptionPane.showMessageDialog(null, "ID 不能为空", "Warning",
JOptionPane.WARNING_MESSAGE);
        }
    }
});
JButton delete = new JButton();
delete.setText("删除");
delete.setFont(new Font("微软雅黑", Font.PLAIN, 18));
delete.setBounds(250, 310, 100, 30);
delete.addActionListener(new ActionListener()
{
    @Override
    public void actionPerformed(ActionEvent e)
    {
        //弹窗，让用户选择是否确认删除学生
        Object[] options = { "是", "否" };
        int option = JOptionPane.showOptionDialog(null, "是否确认删除该学生信
息", "Warning", JOptionPane.DEFAULT_OPTION,
JOptionPane.WARNING_MESSAGE, null, options, options[0]);
    }
}

```

```

        if (option == 0)
        {
            studentlist.remove(sub);//从苏州了表之中移除该学生信息
            //将窗口文本框清空
            IDjtf.setText("");
            Namejtf.setText("");
            Heightjtf.setText("");
            Weightjtf.setText("");
            BMIjtf.setText("");
            PyhsicalConditionjtf.setText("");
            HandleFileTool.saveFile(studentlist, "StudentInfo.txt");//将新数组列表
重新存储到 TXT 文件之中
            JOptionPane.showMessageDialog(null, "删除学生信息成功",
"Message", JOptionPane.INFORMATION_MESSAGE);
        }

    }

});

JButton modify = new JButton();
modify.setText("修改");
modify.setFont(new Font("微软雅黑", Font.PLAIN, 18));
modify.setBounds(120, 310, 100, 30);
modify.addActionListener(new ActionListener()
{
    @Override
    public void actionPerformed(ActionEvent e)
    {
        String id = IDjtf.getText();
        String name = Namejtf.getText();
        String height = Heightjtf.getText();
        String weight = Weightjtf.getText();
        boolean flag1 = true;
        boolean flag2 = true;
        if (!id.equals("") && !name.equals("") && !height.equals("")
&& !weight.equals(""))
        {
            //修改合法性检测，不能修改学生 ID
            if (!ID.equals(IDjtf.getText()))
            {
                JOptionPane.showMessageDialog(null, "不能修改学号", "Error",
JOptionPane.ERROR_MESSAGE);
            } else
            {

```

```

        try
        {
            //检测用户是否输入了浮点型数据，如果不是则是非法数据
            float height1 = Float.parseFloat(Heightjtf.getText());
            float weight1 = Float.parseFloat(Weightjtf.getText());
            flag1 = true;
            if (!isLegalNum(height1, 1.5f, 2.0f) || !isLegalNum(weight1,
45f, 100f))
            {
                JOptionPane.showMessageDialog(null, "数据不合法，体
重和身高数据不合理", "Warning",
                    JOptionPane.WARNING_MESSAGE);
                flag2 = false;
            }

        } catch (Exception e2)
        {
            flag2 = false;
            JOptionPane.showMessageDialog(null, "数据不合法，体重和
身高要求输入浮点数", "Warning",
                JOptionPane.WARNING_MESSAGE);
        }
        if (flag1 && flag2)
        {
            Student student = new Student(id, name,
Float.parseFloat(height), Float.parseFloat(weight));
            studentlist.set(sub, student);
            HandleFileTool.saveFile(studentlist, "StudentInfo.txt");//将修
改之后的数据重新学鹭岛文件之中
            IDjtf.setText(student.getID());
            Namejtf.setText(student.getName());
            Heightjtf.setText("" + student.getHeight());
            Weightjtf.setText("" + student.getWeight());
            BMIjtf.setText("" + student.getBMI());
            PyhsicalConditionjtf.setText(student.getPhysicalCondition());
            JOptionPane.showMessageDialog(null, "学生信息修改成功",
"Message", JOptionPane.INFORMATION_MESSAGE);
        }
    }
    } else
    {
        JOptionPane.showMessageDialog(null, "学生信息不为空", "ERROR",
JOptionPane.ERROR_MESSAGE);
    }
}

```

```

        }
    });

    this.add(modify);
    this.add(delete);
    this.add(IDjtf);
    this.add(Namejtf);
    this.add(Heightjtf);
    this.add(Weightjtf);
    this.add(IDLabel);
    this.add(NameLabel);
    this.add(HeightLabel);
    this.add(WeightLabel);
    this.add(check);
    this.add(PyhsicalConditionLabel);
    this.add(PyhsicalConditionjtf);
    this.add(BMIjtf);
    this.add(BMILabel);
}

/**
 * 检测输入的数据是否是一个在合法范围之内的浮点数*/
private boolean isLegalNum(float num, float min, float max)
{
    if (num <= max && num >= min)
        return true;
    else
        return false;
}
}

class HandleStudentInfo
{
    /**随机生成两百个学生的信息，存放到数组列表 studentlist 中*/
    public static String[][] genStudents(ArrayList<Student> studentlist)
    {
        String id;// 学生 ID
        String name;// 学生姓名
        float height;// 学生身高
        float weight;// 学生体重
        String[][] data = new String[200][6];
        for (int i = 0; i < 200; i++)
        {

```



```

do
{
    id = RandomGenerateID();// 生成学生 ID
} while (isExists(id, studentlist));// 判断学生的 ID 时候已经存在
name = RandomGenerateName();// 随机生成学生名字
height = nextFloat(1.5f, 2.0f);// 生成指定范围 1.5~2.0 的浮点数
weight = nextFloat(45f, 100f);// 生成指定范围 45~100 的浮点数
// 根据生成的信息生成一个新的 Student 对象实例，并增加到数组列表的末
尾

Student student = new Student(id, name, height, weight);
studentlist.add(student);
data[i] = student.toArray();
}
HandleFileTool.saveFile(studentlist, "StudentInfo.txt");
return data;
}

/**
 * 生成一个指定范围 min~max 的浮点数
 */
public static float nextFloat(final float min, final float max)
{
    float a = min + ((max - min) * new Random().nextFloat());
    return (float) (Math.round(a * 100)) / 100;// (这里的 100 就是 2 位小数点,如果要其它
    位,如 4 位,这里两个 100 改成 10000)
}

/**
 * 随机生成一个 201001~211001 的学生 ID
 */
public static String RandomGenerateID()
{
    Random random = new Random();
    int i = random.nextInt(10000);
    return (201001 + i) + "";
}

/**
 * 随机生成学生的名字
 */
public static String RandomGenerateName()
{
    Random random = new Random();

```

```

        String name = "";
        for (int i = 0; i < 5; i++)// 学生名字长度为 5
        {
            name += (char) (random.nextInt(26) + 'a');// 随机生成一个 0~26 的数，计算其余
            字母 a 的相对偏移量，得出一个 ASCII 码字符，连接成一个字符串
        }
        return UpperFirstLetter(name);
    }
}

```

```

/**
 * 将字符串的首字母改成大写，其余的字母改成小写，并返回
 */
public static String UpperFirstLetter(String name)
{
    name = name.toLowerCase();
    return name.substring(0, 1).toUpperCase() + name.substring(1);// 将字符串的第一个
    字母变成大写，并且返回
}

```

```

/**
 * 判断 ID 是否与已经存在，避免重复
 */
public static boolean isExists(String id, ArrayList<Student> studentlist)
{
    for (int i = 0; i < studentlist.size(); i++)
    {
        if (studentlist.get(i) != null)// 数据合法性判断
        {
            if (id.equals(studentlist.get(i).getID()))
            {
                return true;// ID 重复返回 true
            }
        }
    }
    return false;// ID 不重复返回 false
}
}

```

```

class HandleFileTool
{
    /**从文件中读取信息，存放到数组列表 studentlist 中*/
    public static void readFile(ArrayList<Student> studentlist, String string)
    {
        File file = new File(string);
    }
}

```

```

BufferedReader reader = null;
ArrayList<String> information = new ArrayList<String>();
try
{
    reader = new BufferedReader(new FileReader(file));
    String temp;
    while ((temp = reader.readLine()) != null)//判断是否是文件末尾
    {
        information.add(temp);//把学生信息增加到数组列表 information 中
    }
    reader.close();
} catch (IOException e)
{
    e.printStackTrace();
}

String[][] strings = new String[1][5];
//用 information 数组列表里面的信息创建 student 实例，然后存放到 studentlist 中
for (int i = 0; i < information.size(); i++)
{
    strings[0] = information.get(i).split(",");//以逗号为风格符，将数据分开
    studentlist.add(new Student(strings[0][0], strings[0][1],
Float.parseFloat(strings[0][2]),
        Float.parseFloat(strings[0][3])));
}
}

/**从文件中读取信息，存放到二维数组里面中*/
public static void readFile(String[][] data, String filename)
{
    File file = new File(filename);
    BufferedReader reader = null;
    ArrayList<String> information = new ArrayList<String>();
    try
    {
        reader = new BufferedReader(new FileReader(file));
        String temp;
        while ((temp = reader.readLine()) != null)//判断是否读取到文件末尾
        {
            information.add(temp);//把学生信息增加到数组列表 information 中
        }
        reader.close();
    } catch (IOException e)
    {

```

```

        e.printStackTrace();
    }

    //用 information 数组列表里面的信息放置到二维数组 data 里面
    for (int i = 0; i < information.size(); i++)
    {
        data[i] = information.get(i).split(",");//以逗号未分割符
    }
}

```

/**判断文件里面一共有多少行学生信息*/

```

public int HowManyLine(String fileName)
{
    File file = new File(fileName);
    BufferedReader reader = null;
    int line = 0;
    try
    {
        reader = new BufferedReader(new FileReader(file));
        while (reader.readLine() != null)//判断是否读取到文件末尾
        {
            line++;
        }
        reader.close();
    } catch (IOException e)
    {
        e.printStackTrace();
    }
    finally
    {
        if (reader != null)
        {
            try
            {
                reader.close();
            } catch (IOException e1)
            {
            }
        }
    }
    return line;
}

```

/**将数组列表里面为数据存放到 studentlist 中*/

```

public static void saveFile(ArrayList<Student> studentlist, String filename)
{
    try
    {
        FileOutputStream fileOutputStream = new FileOutputStream(filename, false);
        PrintWriter pWriter = new PrintWriter(fileOutputStream);
        String StudentInfo;
        for (int i = 0; i < studentlist.size(); i++)
        {
            StudentInfo = studentlist.get(i).getID() + "," + studentlist.get(i).getName() +
            ","
            + studentlist.get(i).getHeight() + "," + studentlist.get(i).getWeight()
            + ","
            + studentlist.get(i).getBMI() + "," +
            studentlist.get(i).getPhysicalCondition();
            pWriter.println(StudentInfo); //以逗号未分割符，存放到文件之中
        }
        pWriter.close();
        fileOutputStream.close();
    } catch (IOException e)
    {
        // TODO: handle exception
        e.printStackTrace();
    }
}

```

```

class Student
{
    private String id;// 学生学号
    private String name;// 学生姓名
    private float height;// 学生身高
    private float weight;// 学生体重
    private float bmi;// 学生 bmi 值
    private String physicalCondition;// 学生健康状况

    public Student()// 空构造方法，用于创建对象，调用比较器
    {
    }

    public Student(String id, String name, float height, float weight)// 构造方法，初始化数据
    {
        this.id = id;
        this.name = name;
    }
}

```

```

        this.height = height;
        this.weight = weight;
        bmi = calculateBMI(height, weight); // 计算出根据输入的数据计算出学生 BMI
        physicalCondition = checkHealth(bmi); // 根据 BMI 推算出学生的健康状况
    }

//将学生类实例里面的私有域变成一个一维数组
public String[] toArray()
{
    String[] data = new String[6];
    data[0] = id;
    data[1] = name;
    data[2] = String.valueOf(getHeight());
    data[3] = String.valueOf(getWeight());
    data[4] = String.valueOf(getBMI());
    data[5] = physicalCondition;
    return data;
}

@Override
/** 返回学生的个人信息 */
public String toString()
{
    return id + "\t" + name + "\t" + height + "\t" + weight + "\t" + getBMI() + "\t" +
physicalCondition;
}

/** 计算出学生的 bmi */
private float calculateBMI(float height, float weight)
{
    return weight / (height * height);
}

/** 返回私有域 id, 即学生的学号 */
public String getID()
{
    return id;
}

/** 返回私有域 BMI */
public float getBMI()
{
    return (float) (Math.round(bmi * 100)) / 100;
}

```

```

/** 返回私有域 height, 即学生的身高 */
public float getHeight()
{
    return (float) (Math.round(height * 100)) / 100;
}

/** 返回私有域 weight, 即学生的体重 */
public float getWeight()
{
    return (float) (Math.round(weight * 100)) / 100;
}

/** 返回私有域 name, 即学生的姓名 */
public String getName()
{
    return name;
}

/** 返回私有域 PhysicalCondition, 即学生的健康状况 */
public String getPhysicalCondition()
{
    return physicalCondition;
}

/** 根据学生 bmi 退出其健康状况 */
public String checkHealth(float bmis)
{
    String HealthType = null; // 记录学生的健康类型
    if (bmis < 18.5)
    {
        HealthType = "Underweight"; // 体重过轻
    } else if (18.5 <= bmis && bmis < 23)
    {
        HealthType = "Normal Range"; // 正常范围
    } else if (23 <= bmis && bmis < 25)
    {
        HealthType = "Overweight-At Risk"; // 有肥胖的趋势
    } else if (25 <= bmis && bmis < 30)
    {
        HealthType = "Overweight-Moderately Obese"; // 超重
    } else if (30 <= bmis)
    {
        HealthType = "Overweight-Severely Obese"; // 严重超重
    }
}

```

```

        } else
        {
        }
        return HealthType;
    }

/**
 * Comparator 接口，重写 compare 方法，实现按 ID 值对学生进行升序排序
 */
class SortByIDAsc implements Comparator<Student>
{
    public int compare(Student o1, Student o2)
    {
        return o1.getID().compareTo(o2.getID());
    }
}

/**
 * Comparator 接口，重写 compare 方法，实现按 ID 值对学生进行降序排序
 */
class SortByIDDsc implements Comparator<Student>
{
    public int compare(Student o1, Student o2)
    {
        return o2.getID().compareTo(o1.getID());
    }
}

/**
 * Comparator 接口，重写 compare 方法，实现按姓名值对学生进行升序排序
 */
class SortByNameAsc implements Comparator<Student>
{
    public int compare(Student s1, Student s2)
    {
        return s1.getName().compareTo(s2.getName());
    }
}

/**
 * Comparator 接口，重写 compare 方法，实现按姓名值对学生进行降序排序
 */
class SortByNameDsc implements Comparator<Student>
{

```



```

        public int compare(Student s1, Student s2)
        {
            return s2.getName().compareTo(s1.getName());
        }
    }

/**
 * Comparator 接口，重写 compare 方法，实现按身高值对学生进行升序排序
 */
class SortByHeightAsc implements Comparator<Student>
{
    public int compare(Student s1, Student s2)
    {
        if (s1.getHeight() < s2.getHeight())
            return -1;
        return 1;
    }
}

/**
 * Comparator 接口，重写 compare 方法，实现按身高值对学生进行降序排序
 */
class SortByHeightDsc implements Comparator<Student>
{
    public int compare(Student s1, Student s2)
    {
        if (s2.getHeight() < s1.getHeight())
            return -1;
        return 1;
    }
}

/**
 * Comparator 接口，重写 compare 方法，实现按体重值对学生进行升序排序
 */
class SortByWeightAsc implements Comparator<Student>
{
    public int compare(Student s1, Student s2)
    {
        if (s1.getWeight() < s2.getWeight())
            return -1;
        return 1;
    }
}

```

```

/**
 * Comparator 接口，重写 compare 方法，实现按体重值对学生进行降序排序
 */
class SortByWeightDsc implements Comparator<Student>
{
    public int compare(Student s1, Student s2)
    {
        if (s2.getWeight() < s1.getWeight())
            return -1;
        return 1;
    }
}

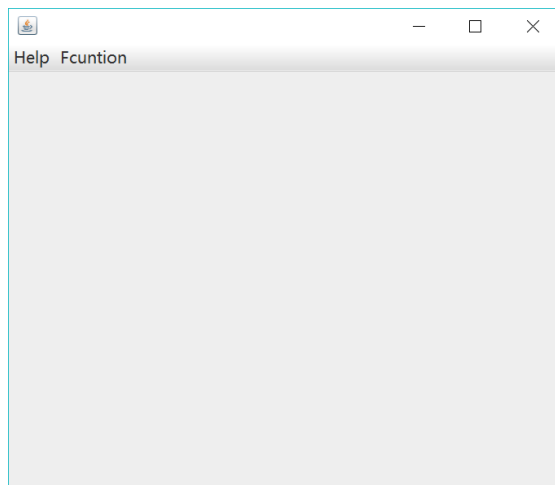
/**
 * Comparator 接口，重写 compare 方法，实现按 BMI 值对学生进行升序排序
 */
class SortByBMIAsc implements Comparator<Student>
{
    public int compare(Student s1, Student s2)
    {
        if (s1.getBMI() < s2.getBMI())
            return -1;
        return 1;
    }
}

/**
 * Comparator 接口，重写 compare 方法，实现按 BMI 值对学生进行降序排序
 */
class SortByBMIDsc implements Comparator<Student>
{
    public int compare(Student s1, Student s2)
    {
        if (s2.getBMI() < s1.getBMI())
            return -1;
        return 1;
    }
}
}

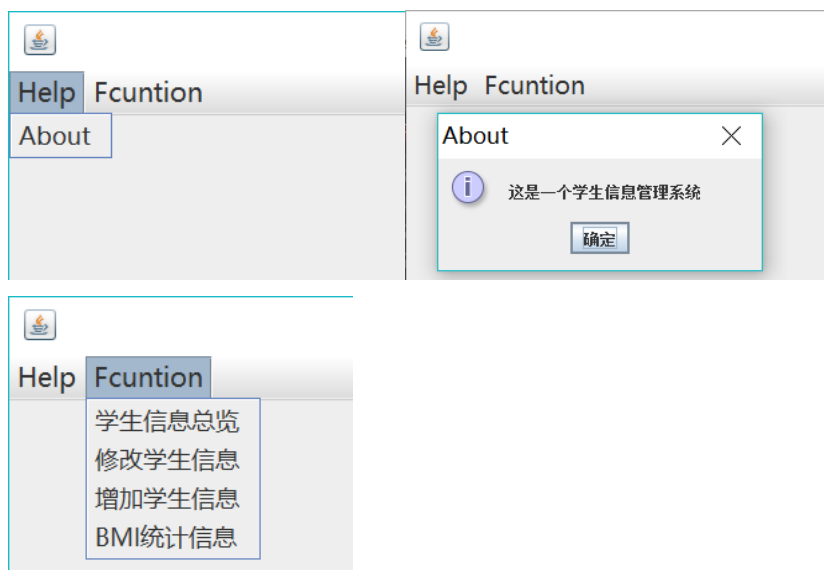
```

程序运行结果：

很简陋的主界面：



菜单：



功能一：学生信息总览，增加学生信息

1、总览学生信息

Help Fcuntion					
增加数据		请选择排序方式			
学号	姓名	身高	体重	BMI值	健康状况
206288	Yxlud	1.98	54.76	13.97	Underweight
209341	Qxbus	1.63	71.53	26.92	Overweight-Moderately Obese
210361	Ckwra	1.83	79.46	23.73	Overweight-At Risk
203311	Lkbym	1.93	77.75	20.87	Normal Range
205341	Zmczj	1.99	65.46	16.53	Underweight
209130	Xceqm	1.9	70.36	19.49	Normal Range
203176	Xhxtt	1.99	98.62	24.9	Overweight-At Risk
205546	Nricm	1.54	68.22	28.77	Overweight-Moderately Obese
210697	Ljqpt	1.71	85.2	29.14	Overweight-Moderately Obese
207546	Tnack	1.81	89.97	27.46	Overweight-Moderately Obese
206269	Gqrdn	1.92	46.72	12.67	Underweight

2、排序:

请选择排序方式
请选择排序方式
按学号从小到大排序
按姓名从小到大排序
按身高从小到大排序
按体重从小到大排序
按BMI值从小到大排序

① 按学号大小从小到大排序

Help Fcuntion					
增加数据		按学号从小到大排序			
学号	姓名	身高	体重	BMI值	健康状况
203176	Xhxtt	1.99	98.62	24.9	Overweight-At Risk
203311	Lkbym	1.93	77.75	20.87	Normal Range
205341	Zmczj	1.99	65.46	16.53	Underweight
205546	Nricm	1.54	68.22	28.77	Overweight-Moderately Obese
206269	Gqrdn	1.92	46.72	12.67	Underweight
206288	Yxlud	1.98	54.76	13.97	Underweight
207546	Tnack	1.81	89.97	27.46	Overweight-Moderately Obese
209130	Xceqm	1.9	70.36	19.49	Normal Range
209341	Qxbus	1.63	71.53	26.92	Overweight-Moderately Obese
210361	Ckwra	1.83	79.46	23.73	Overweight-At Risk
210697	Ljqpt	1.71	85.2	29.14	Overweight-Moderately Obese

② 按姓名大小从小到大排序

Help Fcuntion					
增加数据					
按姓名从小到大排序					
学号	姓名	身高	体重	BMI值	健康状况
210361	Ckwra	1.83	79.46	23.73	Overweight-At Risk
206269	Gqrdn	1.92	46.72	12.67	Underweight
210697	Ljqpt	1.71	85.2	29.14	Overweight-Moderately Obese
203311	Lkbym	1.93	77.75	20.87	Normal Range
205546	Nricm	1.54	68.22	28.77	Overweight-Moderately Obese
209341	Qxbus	1.63	71.53	26.92	Overweight-Moderately Obese
207546	Tnack	1.81	89.97	27.46	Overweight-Moderately Obese
209130	Xceqm	1.9	70.36	19.49	Normal Range
203176	Xhxtt	1.99	98.62	24.9	Overweight-At Risk
206288	Yxlud	1.98	54.76	13.97	Underweight
205341	Zmczj	1.99	65.46	16.53	Underweight

③ 按身高大小从小到大排序

Help Fcuntion					
增加数据					
按身高从小到大排序					
学号	姓名	身高	体重	BMI值	健康状况
205546	Nricm	1.54	68.22	28.77	Overweight-Moderately Obese
209341	Qxbus	1.63	71.53	26.92	Overweight-Moderately Obese
210697	Ljqpt	1.71	85.2	29.14	Overweight-Moderately Obese
207546	Tnack	1.81	89.97	27.46	Overweight-Moderately Obese
210361	Ckwra	1.83	79.46	23.73	Overweight-At Risk
209130	Xceqm	1.9	70.36	19.49	Normal Range
206269	Gqrdn	1.92	46.72	12.67	Underweight
203311	Lkbym	1.93	77.75	20.87	Normal Range
206288	Yxlud	1.98	54.76	13.97	Underweight
203176	Xhxtt	1.99	98.62	24.9	Overweight-At Risk
205341	Zmczj	1.99	65.46	16.53	Underweight

④ 按体重大小从小到大排序

Help Fcuntion					
增加数据					
按体重从小到大排序					
学号	姓名	身高	体重	BMI值	健康状况
206269	Gqrdn	1.92	46.72	12.67	Underweight
206288	Yxlud	1.98	54.76	13.97	Underweight
205341	Zmczj	1.99	65.46	16.53	Underweight
205546	Nricm	1.54	68.22	28.77	Overweight-Moderately Obese
209130	Xceqm	1.9	70.36	19.49	Normal Range
209341	Qxbus	1.63	71.53	26.92	Overweight-Moderately Obese
203311	Lkbym	1.93	77.75	20.87	Normal Range
210361	Ckwra	1.83	79.46	23.73	Overweight-At Risk
210697	Ljqpt	1.71	85.2	29.14	Overweight-Moderately Obese
207546	Tnack	1.81	89.97	27.46	Overweight-Moderately Obese
203176	Xhxtt	1.99	98.62	24.9	Overweight-At Risk

⑤ 按 BMI 值大小从小到大排序

Help Fcuntion					
		增加数据	按BMI值从小到大排序		
学号	姓名	身高	体重	BMI值	健康状况
206269	Gqrdn	1.92	46.72	12.67	Underweight
206288	Yxlud	1.98	54.76	13.97	Underweight
205341	Zmczj	1.99	65.46	16.53	Underweight
209130	Xceqm	1.9	70.36	19.49	Normal Range
203311	Lkbym	1.93	77.75	20.87	Normal Range
210361	Ckwra	1.83	79.46	23.73	Overweight-At Risk
203176	Xhxtt	1.99	98.62	24.9	Overweight-At Risk
209341	Qxbus	1.63	71.53	26.92	Overweight-Moderately Obese
207546	Tnack	1.81	89.97	27.46	Overweight-Moderately Obese
205546	Nricm	1.54	68.22	28.77	Overweight-Moderately Obese
210697	Ljqpt	1.71	85.2	29.14	Overweight-Moderately Obese

3、增加 200 名学生

Help Fcuntion					
		增加数据	按BMI值从小到大排序		
学号	姓名	身高	体重	BMI值	健康状况
207025	Cihwn	1.98	45.2	11.53	Underweight
209997	Tjldz	1.95	46.19	12.15	Underweight
206078	Zabfb	1.98	47.93	12.23	Underweight
206269	Gqrdn	1.92	46.72	12.67	Underweight
203238	Hyrfx	1.89	45.68	12.79	Underweight
207850	Tkaau	1.95	49.98	13.14	Underweight
204474	Xyngv	1.89	47.63	13.33	Underweight
203609	Oaekl	1.85	46.91	13.71	Underweight
206755	Zcbva	1.83	46.11	13.77	Underweight
206288	Yxlud	1.98	54.76	13.97	Underweight
202041	Gkszv	1.99	55.49	14.01	Underweight
201395	Ywwac	1.9	50.77	14.06	Underweight
207763	Dglav	1.94	53.49	14.21	Underweight
210955	Fhtlm	1.96	54.61	14.22	Underweight
210855	Climq	1.8	46.1	14.23	Underweight
210939	Crumw	1.89	51.22	14.34	Underweight
201478	Bzrrc	1.95	54.75	14.4	Underweight
205184	Lriqd	1.79	47.51	14.83	Underweight
208973	Ugznm	1.89	53.15	14.88	Underweight
201266	Wwywt	1.87	52.54	15.02	Underweight
206760	Vdqik	1.77	48.34	15.43	Underweight
202856	Yckjh	1.75	47.93	15.65	Underweight
203604	Skvuh	1.72	48.16	16.28	Underweight
207815	Ehdzv	1.71	48.31	16.52	Underweight
205341	Zmczj	1.99	65.46	16.53	Underweight
210477	lhkn	1.75	50.92	16.63	Underweight
204056	Pmgrp	1.94	63.19	16.79	Underweight
202621	Ajhrr	1.98	66.21	16.89	Underweight
205219	Hkzdu	1.98	66.94	17.07	Underweight
206694	Taofg	1.78	54.15	17.09	Underweight
208522	Wdhel	1.69	48.95	17.14	Underweight
204359	Wvibi	1.86	59.49	17.2	Underweight
208503	Ocpdg	1.73	51.78	17.3	Underweight
206996	Xdgri	2.0	69.35	17.34	Underweight
204808	Rclfh	1.75	53.58	17.5	Underweight
209483	Gzlvh	1.8	56.7	17.5	Underweight
204799	Loevn	1.91	64.15	17.58	Underweight
205174	Aytkj	1.82	58.38	17.62	Underweight
205032	Zarns	1.92	65.34	17.72	Underweight
203315	Hskmf	1.97	68.9	17.75	Underweight
207566	Igjps	1.88	62.81	17.77	Underweight
203799	Rpexn	1.9	64.61	17.9	Underweight
206155	Mhooj	1.65	48.75	17.91	Underweight
210488	Vsonq	1.76	55.68	17.98	Underweight
201384	Afqdh	1.69	51.42	18.0	Underweight
205780	Ktfwh	1.9	65.68	18.19	Underweight
204379	Wntr	1.75	55.81	18.22	Underweight
202333	Woujj	1.6	46.79	18.28	Underweight

功能二：维护学生信息

1、修改学生信息（按学号修改）



Help Fcuntion

ID: 203063 查询

姓名:

身高:

体重:

BMI值:

健康状况:

修改 删除



Help Fcuntion

ID: 203063 查询

姓名: Ovfnv

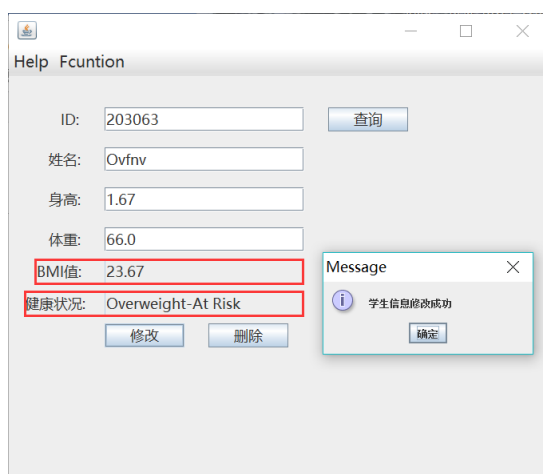
身高: 1.67

体重: 84.48

BMI值: 30.29

健康状况: Overweight-Severely Obese

修改 删除



Help Fcuntion

ID: 203063 查询

姓名: Ovfnv

身高: 1.67

体重: 66.0

BMI值: 23.67

健康状况: Overweight-At Risk

修改 删除

Message

学生信息修改成功

确定

修改成功，学生的 BMI 值与健康状况都会随之更新

2、删除学生

Help

Fcuntion

ID:

203063

查询

姓名:

Ovfnv

身高:

1.67

体重:

66.0

BMI值:

23.67

健康状况:

Overweight-At Risk

修改

删除

Warning

是否确认删除该学生信息

是

否

Help

Fcuntion

ID:

查询

姓名:

身高:

体重:

BMI值:

健康状况:

修改

删除

Message

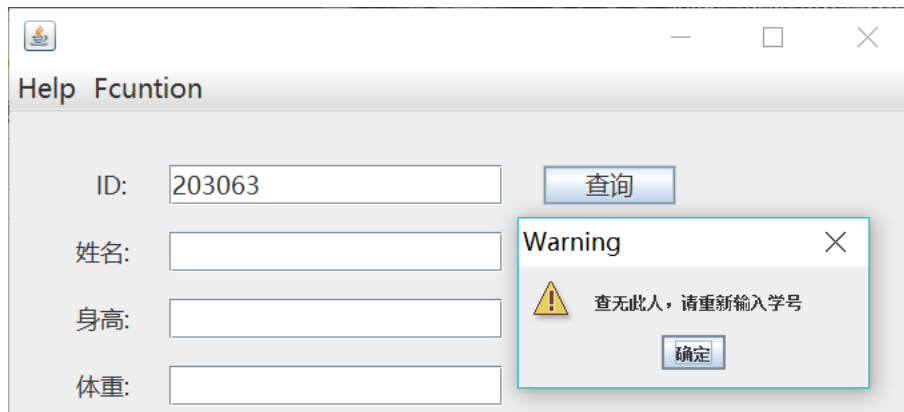
i

删除学生信息成功

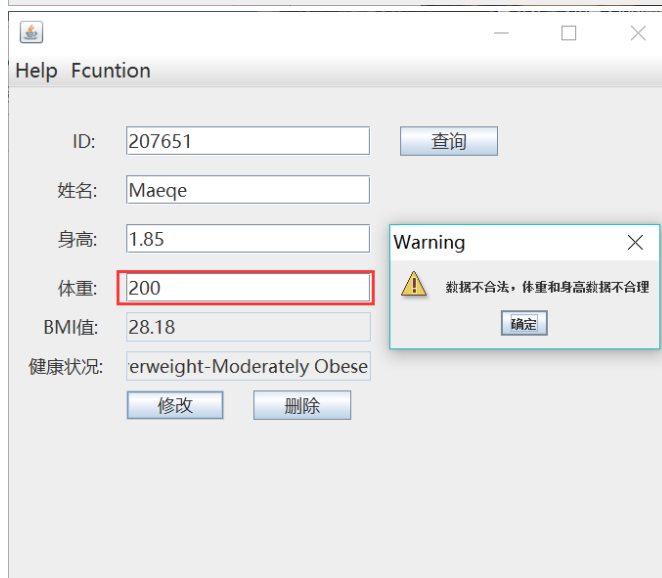
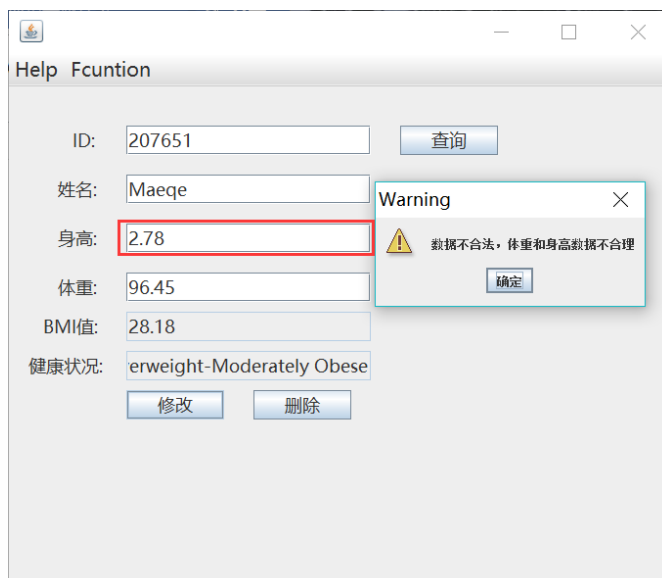
确定

3、错误处理:

① 想要修改的学生信息不存在



② 体重，身高数据不合法



③ 修改学号会报错（学号不能修改）

Help Fcuntion

ID: 207652 查询

姓名: Maeqe

身高: 1.85

体重: 96.45

BMI值: 28.18

健康状况: erweight-Moderately Obese

修改 删除

Error

不能修改学号

确定

功能三、增加学生信息

1、输入数据，进行合法性检测，然后增加学生信息

Help Fcuntion

ID: 201001

姓名: Jack

身高: 1.78

体重: 72

BMI值:

健康状况:

增加 合法性检测

Message

数据合法

确定

Help Fcuntion

ID: 201001

姓名: Jack

身高: 1.78

体重: 72

BMI值: 22.72

健康状况: Normal Range

增加 合法性检测

Message

增加学生信息成功

确定

2、错误处理

① 在进行数据合法性检测前，“增加”这个选项一直处于 `setEnabled(false)` 状态

② ID 已经被占用

Help Fcuntion

ID: 204993

姓名: Marry

身高: 1.72

体重: 65

BMI值:

健康状况:

增加 合法性检测

ERROR

ID已经被占用，请重新输入ID

确定

③ 输入数据不合法

Help Fcuntion

ID: 204994

姓名: Marry

身高: 23

体重: 65

BMI值:

健康状况:

增加

合法性检测

Warning

!

数据不合法，体重和身高数据不合理

确定

Help Fcuntion

ID: 204994

姓名: Marry

身高: 1.72

体重: 200

BMI值:

健康状况:

增加

合法性检测

Warning

!

数据不合法，体重和身高数据不合理

确定

功能四：BMI 值统计信息总览

