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

实验四：面向对象程序设计

一、实验目的

1）掌握面向对象的基本概念（成员变量、成员函数等）
2）掌握类的定义、内部类的定义
3）掌握对象的声明
4）掌握对象数组的使用
5）基本算法的设计

二、实验内容

1）编写 OOBMI 类文件；并在该类文件中定义另一个类 Student，该类包含学号、姓名、身高、体重和 bmi 等属性，为 Student 类定义创建函数。

2）在 Student 类中，增加 public String toString()函数，该函数可以返回一个字符串，该字符串包含学号、姓名、身高、体重、bmi 值和胖瘦健康状况，他们之间用制表符(\t)隔开；

3）在 OOBMI 中增加成员属性 Student[] students；在 OOBMI 中增加 genStudents 函数，参数为整数，能够随机生成指定数量的名学生对象，并保存到 students 数组中。（注意，学号、姓名、身高、体重等均需随机生成，数值均需保留两位小数存储）

4）在 OOBMI 类中增加 public boolean isExists(String id) 函数，判断该学生是否已经在 students 数组中，函数返回值为 boolean 类型，如果已经存在，返回 false；否则，返回 true。并在 genStudents 函数，调用 isExists 函数避免输入或生成重复的学号的学生。

5）在 OOBMI 中增加 4 个函数，分别统计 bmi 的均值、中值、众数、方差等统计信息。

6）在 OOBMI 中增加 printStatics 函数，该函数首先打印所有学生基本信息，然后打印 bmi 的均值、中值、众数、方差等统计结果信息。

7）增加 menu 函数提供随机生成学生、打印学生、5 种排序、打印统计信息、退出执行等 9 个菜单功能，用户输入指定选项后，运行相应函数功能。

8）在 OOBMI 的 main 函数中，调用 menu 函数，测试运行各项功能。

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、实验代码

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，注意源程序于报告请分别上传到不同的文件夹中！

程序代码：

```
package edu.hit.java.exp2.hit1173710105;

import java.util.*;

public class OOBMI
{
```

```
public static void main(String[] args)
{
    Student[] students = null;//创建学生数组,用于存储学生信息
    Scanner scanner = new Scanner(System.in);
    System.out.print("请输入学生数目:");
    int num = scanner.nextInt();//记录学生数目
    int i;
    int[] sortedIndex = new int[num];//  该数组中保存了进行排序的数组排序后的下标
    for(int j = 0;j<sortedIndex.length;j++)
    {
        sortedIndex[j] = j;//默认顺序
    }
    do
    {
        i = menu();//  获取用户想执行功能
        switch (i)
        {
        case 1:
            //生成学生信息
            students = genStudents(num);
            break;
        case 2:
            //打印学生信息
            OOBMI.printStudents(students, sortedIndex);
            break;
        case 3:
            //按学号从小到大排序
            OOBMI.sortByID(students, sortedIndex);
            break;
        case 4:
            //按姓名从小到大排序
            OOBMI.sortByName(students, sortedIndex);
            break;
        case 5:
            //按身高从小到大排序
            OOBMI.sortByHeights(students, sortedIndex);
            break;
        case 6:
            //按体重从小到大排序
            OOBMI.sortByWeights(students, sortedIndex);
            break;
        case 7:
            //按 BMI 从小到大排序
            OOBMI.sortByBMI(students, sortedIndex);
```

```
        break;
case 8:
        //打印菜单
        System.out.println("1.查看学生 BMI 的均值");
        System.out.println("2.查看学生 BMI 的中值");
        System.out.println("3.查看学生 BMI 的众数");
        System.out.println("4.查看学生 BMI 的方差");
        System.out.print("请选择:");
        int option = scanner.nextInt();
        float result = 0;
        if(option==1)
        {
            //计算学生 BMI 的平均值
            result = calculateAverageOfBMI(students);
            if(result!=0f)
                System.out.println("学生 BMI 的平均值为:"+String.format("%.2f",result));
        }
        else if(option==2)
        {
            //计算学生 BMI 的中位数
            result = calculateMedian(students);
            if(result!=0f)
                System.out.println("学生 BMI 的中位数为:"+String.format("%.2f",result));
        }
        else if(option==3)
        {
            //计算学生 BMI 的众数
            findMode(students);
        }
        else if(option==4)
        {
            //计算学生 BMI 的方差
            result = calculateVariance(students);
            if(result != 0f)
                System.out.println("学生的 BMI 的方差为:"+String.format("%.2f", result));
        }

        else {}
        break;
case 9:
        //程序结束，退出
        System.out.println("It is the end!");
        scanner.close();
        System.exit(0);
```

```
                    break;
                default:
                    break;
                }
            } while (true);
        }

        /**
         * 随机生成指定数量的学生信息*/
        public static Student[] genStudents(int num)
        {
            Student[] students = new Student[num];//创建学生数组，存储学生信息
            String id;//学生 ID
            String name;//学生姓名
            float height;//学生身高
            float weight;//学生体重
            for(int i=0;i<num;i++)
            {
                do
                {
                    id = String.valueOf(201001+i);//生成学生 ID
                }while(isExists(id, students));//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 455~100 的浮点数
                students[i] = new Student(id, name, height, weight);//数组每一个元素都指向一个实例化的学
生对象
            }
            return students;
        }
        /**
         * 生成一个指定范围 min~max 的浮点数*/
        public static float nextFloat(final float min, final float max)
        {
            return min + ((max - min) * new Random().nextFloat());
        }
        /**
         * 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 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,Student[] students)
    {
        for(int i=0;i<students.length;i++)
        {
            if(students[i]!=null)//数据合法性判断
            {
                if(id.equals(students[i].getID()))
                {
                    return true;//ID 重复返回 true
                }
            }
        }
        return false;//ID 不重复返回 false
    }

    public static float[] getBMIS(Student[] students)
    {
        float[] bmis = new float[students.length];
        for(int i =0;i<students.length;i++)
        {
            bmis[i] = students[i].getBMI();
        }
        return bmis;
    }

    /**
     * 将所有的学生的实例化对象的 id 属性取出，放到一个数组里*/
    public static String[] getIDS(Student[] students)
    {
        String[] IDS = new String[students.length];
        for(int i =0;i<students.length;i++)
        {
            IDS[i] = students[i].getID();
```

```
        }
        return IDS;
}

/**
 * 将所有的学生的实例化对象的 name 属性取出，放到一个数组里*/
public static String[] getName(Student[] students)
{
        String[] Name = new String[students.length];
        for(int i =0;i<students.length;i++)
        {
                Name[i] = students[i].getName();
        }
        return Name;
}

/**
 * 将所有的学生的实例化对象的 height 属性取出，放到一个数组里*/
public static float[] getHeight(Student[] students)
{
        float[] height = new float[students.length];
        for(int i =0;i<students.length;i++)
        {
                height[i] = students[i].getHeight();
        }
        return height;
}

/**
 * 将所有的学生的实例化对象的 weight 属性取出，放到一个数组里*/
public static float[] getWeight(Student[] students)
{

        float[] Weight = new float[students.length];
        for(int i =0;i<students.length;i++)
        {
                Weight[i] = students[i].getWeight();
        }
        return Weight;
}

/**
 * 将所有的学生的实例化对象的 PhysicalCondition 属性取出，放到一个数组里*/
public static String[] getPhysicalCondition(Student[] students)
```

```
{
        String[] PhysicalCondition = new String[students.length];
        for(int i =0;i<students.length;i++)
        {
                PhysicalCondition[i] = students[i].getPhysicalCondition();
        }
        return PhysicalCondition;
}

/**
 * 计算所有学生的 BMI 的平均值*/
public static float calculateAverageOfBMI(Student[] students)
{
        if (students==null)//判断是否是非法操作
        {
                System.out.println("非法操作，请输入学生信息之后再进行此项操作");
                return 0;
        }
        float average = 0;
        for(int i = 0;i<students.length;i++)
        {
                average += students[i].getBMI();//迭代累加
        }
        return average/students.length;
}

/**
 * 计算所有学生的 BMI 的中位数*/
public static float calculateMedian(Student[] students)
{
        if (students==null)//判断是否是非法操作
        {
                System.out.println("非法操作，请输入学生信息之后再进行此项操作");
                return 0;
        }
        float[] bmis = getBMIS(students);//取出所有学生的 BMI 值，放到数组 bmis 中
        SelectionSort(bmis);//将数组 bmi 从小到大进行排序
        float Median;
        if(bmis.length%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
length/2 的值
        {
                Median = bmis[bmis.length/2];
        }
```

```
        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 length/2 与 length/2+1 的值的平均值
        {
            Median =    (bmis[bmis.length/2+1]+bmis[bmis.length/2])/2;
        }
        return Median;
    }

    /**
     * 选择排序法，对数组进行从小到大排序*/
    public static void SelectionSort(float[] bmis)
    {
        if (bmis==null||bmis[0]==0)//判断是否是非法操作
        {
            System.out.println("非法操作，请输入学生信息之后再进行此项操作");
        }

        float minValue;//最小值
        int minSub;//最小值得小标
        float temp;//暂时变量
        for(int i=0;i<bmis.length-1;i++)
        {
            minValue = bmis[i];//假设最小值为 bmis[i]
            minSub = i;//保存假设值得下标
            for(int j=i+1;j<bmis.length;j++)
            {
                if(bmis[j]<=minValue)//寻找最小值，并保存其值与下标
                {
                    minSub = j;
                    minValue = bmis[j];
                }
            }
            if(minSub!=i)//如果为 false，则表示存在比假设值更小的数
            {
                //交换
                temp = bmis[i];
                bmis[i] = minValue;
                bmis[minSub] = temp;
            }

        }
    }

    /**
```

```
 * 用于寻找 student 对象数组里面所有学生各自 BMI 值的众数*/
public static void findMode(Student[] students)
{

    if (students==null)//判断是否是非法操作
    {
        System.out.println("非法操作，请输入学生信息之后再进行此项操作");
        return;
    }

    float[] bmis = getBMIS(students);//获得 student 对象数组里面每个学生的 BMI 值
    float[] printBMIS = bmis.clone();//用于打印原来数值
    int length = bmis.length;//记录学生的个数，即数组 BMI 的长度
    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

    SelectionSort(bmis);//先将数组 bmis 排序

    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++;
        }
}

//打印数据信息

System.out.print("\nBMI 众数统计情况:");
for(int i =0;i<length;i++)
{
        if(num[i]!=0)//过滤非法数据
        {
                System.out.print(String.format("%.2f",num[i])+"\t");
        }
}

System.out.print("\n"+"每个数值出现次数:");
for(int i=0;i<length;i++)
{
        if(count[i]!= 0f)//过滤非法数据
        {
                System.out.printf(String.format("%5d", count[i])+" \t");
        }
}

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("\n"+"众数不存在！ ");
        return;
```

```
        }

        System.out.print("\n"+"众数为:");
        for(int i=0;i<length;i++)
        {
            if(ModeSub[i]!=0)//过滤非法数据
            {
                System.out.print(String.format("%.2f",num[ModeSub[i]])+"\t");
            }
        }
}

/**
 *计算所有学生的 BMI 的方差*/
public static float calculateVariance(Student[] students)
{
        if (students==null)//判断是否是非法操作
        {
            System.out.println("非法操作，请输入学生信息之后再进行此项操作");
            return 0;
        }
        float[] bmis = getBMIS(students);//获得 student 对象数组里面每个学生的 BMI 值
        float average = calculateAverageOfBMI(students);//计算所有学生的 BMI 平均值
        float sum = 0;
        int length = bmis.length;
        for(int i = 0;i<length;i++)
        {
            sum += (bmis[i]-average)*(bmis[i]-average);//迭代计算出方差
        }
        return sum/length;
}

/**
 * 打印菜单信息*/
public static void printMenuMessage()
{
        System.out.println("本程序功能： ");
        System.out.println("1.随机生成学生信息");
        System.out.println("2.打印学生信息");
        System.out.println("3.按学号进行小到大排序");
        System.out.println("4.按姓名进行小到大排序");
        System.out.println("5.按身高进行小到大排序");
        System.out.println("6.按体重进行小到大排序");
        System.out.println("7.按 BMI 进行小到大排序");
```

```
        System.out.println("8.查看学生 BMI 统计信息");
        System.out.println("9.退出程序");
}

/**
 * 打印菜单，并且获取用户输入
 * 返回这个输入所对应的功能的索引*/
public static int menu()
{
        String num = null;// 获取用户想执行功能
        int i = 8;// 获取用户想执行功能
        boolean flag = true;// 错误输入标志;
        Scanner scanner = new Scanner(System.in);
        do
        {
                if (flag)// 只打印一次菜单，避免重复
                {
                        OOBMI.printMenuMessage();
                }
                num = scanner.nextLine();
                // 检查非法输入，如果用户输入非数字，这再次输入
                try
                {
                        i = Integer.parseInt(num);
                        flag = true;
                }
                catch (Exception e)
                {
                        flag = false;
                }
                if (!flag)
                {
                        System.out.println("输入有误，请重新输入:");
                }
        } while (!flag || i > 9 || i < 0);
        return i;
}

/**
 * 按格式，按顺序打印学生信息*/
public static void printStudents(Student[] students,int[] sortedIndex)
{
        if (students[0].getID() == null || "".equals(students[0].getID()))//判断是否是非法操作
        {
```

```
            System.out.println("非法操作，请输入学生信息之后再进行此项操作");
            return;
        }

        String[] ids = getIDS(students);//学生 ID
        String[] names = getName(students);//学生姓名
        float[] heights = getHeight(students);//学生身高
        float[] weights = getWeight(students);//学生体重
        float[] bmis = getBMIS(students);//学生 BMI
        String[] PhysicalCondition = getPhysicalCondition(students);//学生健康状况

        System.out.println("学生信息总览:");
        System.out.println("学号" + "\t" + "姓名" + "\t" + "  身高" + "\t" + "  体重" + "\t" + "BMI 指数" + "\t"
+ "      健康状况");
        //sortedIndex 数组里面存储了经从小大排序以后的数组下标
        //按这个小标打印学生信息
        for (int i = 0; i < sortedIndex.length; i++)
        {
            System.out.print(ids[sortedIndex[i]] + "\t");//打印学号
            System.out.print(names[sortedIndex[i]] + "\t");//打印姓名
            System.out.print(String.format("%.2f",heights[sortedIndex[i]])+"\t");//打印身高
            System.out.print(String.format("%.2f",weights[sortedIndex[i]]) + "\t");//打印体重
            System.out.print(String.format("%.2f",bmis[sortedIndex[i]]) + "\t");//打印 BMI 的值
            System.out.println(PhysicalCondition[sortedIndex[i]]);//打印健康状况
        }
    }

    /**
     * 按 ID 从小到大排序，将排序结果存放到数组 sortedIndex 里面*/
    public static void sortByID(Student[] students, int[] sortedIndex)
    {
        if (students[0].getID() == null || "".equals(students[0].getID()))//判断是否是非法操作
        {
            System.out.println("非法操作，请输入学生信息之后再进行此项操作");
            return;
        }

        String[] ids = getIDS(students);
        String minValue = "9999999999";// 记录最小值
        int minSub = 0;// 记录最小值的下标
        int k = 0;// 用于 sortedIndex 下标自增
        for (int j = 0; j < ids.length; j++)
        {
            for (int i = 0; i < ids.length; i++)
```

```
            {
                if (ids[i] != null)//判断参与比较的数据是否是合法的，如果不合法，就剔除掉，不参与比较
                {
                    if (ids[i].compareTo(minValue) <= 0)//在数组里面找最小值，并且记录下标
                    {
                        minValue = ids[i];//迭代最小值
                        minSub = i;//记录当前最小值下标
                    }
                }
            }
            ids[minSub] = null;//将数组里面最小的值剔除，不再参与比较
            minValue = "9999999999";//重设最小值
            sortedIndex[k] = minSub;//将最小值下标按顺序记录在数组 sortedIndex 中
            k++;
        }
    }

    /**
     * 按姓名从小到大排序，将排序结果存放到数组 sortedIndex 里面*/
    public static void sortByName(Student[] students, int[] sortedIndex)
    {
        if (students[0].getName() == null || "".equals(students[0].getName()))//判断是否是非法操作
        {
            System.out.println("非法操作，请输入学生信息之后再进行此项操作");
        }

        String[] names = getName(students);
        String minValue = "zzzzzzzz";//  记录最小值
        int minSub = 0;//  记录最小值的下标
        int k = 0;//  用于 sortedIndex 下标自增
        for (int j = 0; j < names.length; j++)
        {
            for (int i = 0; i < names.length; i++)
            {
                if (names[i] != null)//判断参与比较的数据是否是合法的，如果不合法，就剔除掉，不参与比较
                {
                    if (names[i].compareTo(minValue) <= 0)//在数组里面找最小值，并且记录下标
                    {
                        minValue = names[i];//迭代最小值
                        minSub = i;//记录当前最小值下标
                    }
                }
```

```
                }
                names[minSub] = null;//将数组里面最小的值剔除，不再参与比较
                minValue = "zzzzzzzz";//重设最小值
                sortedIndex[k] = minSub;//将最小值下标按顺序记录在数组 sortedIndex 中
                k++;
            }
        }

        /**
         * 按身高从小到大排序，将排序结果存放到数组 sortedIndex 里面*/
        public static void sortByHeights(Student[] students, int[] sortedIndex)
        {
            if (students[0].getHeight() == 0f)//判断是否是非法操作
            {
                System.out.println("非法操作，请输入学生信息之后再进行此项操作");
            }
            float[] heights = getHeight(students);
            float minValue = 10f;// 记录最小值
            int minSub = 0;// 记录最小值的下标
            int k = 0;// 用于 sortedIndex 下标自增
            for (int j = 0; j < heights.length; j++)
            {
                for (int i = 0; i < heights.length; i++)
                {
                    if (heights[i] > 0 )//判断参与比较的数据是否是合法的，如果不合法，就剔除掉，不参
与比较
                    {
                        if (heights[i] <= minValue)//在数组里面找最小值，并且记录下标
                        {
                            minValue = heights[i];//迭代最小值
                            minSub = i;//记录当前最小值下标
                        }
                    }
                }
                heights[minSub] = -1f;//将数组里面最小的值剔除，不再参与比较
                minValue = 10;//重设最小值
                sortedIndex[k] = minSub;//将最小值下标按顺序记录在数组 sortedIndex 中
                k++;
            }
        }

        /**
         *按 BMI 值从小到大排序，将排序结果存放到数组 sortedIndex 里面 */
        public static void sortByBMI(Student[] students, int[] sortedIndex)
```

```
{
    if (students[0].getBMI() == 0f)//判断是否是非法操作
    {
        System.out.println("非法操作，请输入学生信息之后再进行此项操作");
    }

    float[] bmis = getBMIS(students);
    float minValue = 100f;// 记录最小值
    int minSub = 0;// 记录最小值的下标
    int k = 0;// 用于 sortedIndex 下标自增
    for (int j = 0; j < bmis.length; j++)
    {
        for (int i = 0; i < bmis.length; i++)
        {
            if (bmis[i] > 0)//判断参与比较的数据是否是合法的，如果不合法，就剔除掉，不参与
比较
            {
                if (bmis[i] <= minValue)//在数组里面找最小值，并且记录下标
                {
                    minValue = bmis[i];//迭代最小值
                    minSub = i;//记录当前最小值下标
                }
            }
        }
        bmis[minSub] = -1f;//将数组里面最小的值剔除，不再参与比较
        minValue = 100f;//重设最小值
        sortedIndex[k] = minSub;//将最小值下标按顺序记录在数组 sortedIndex 中
        k++;
    }
}

/**按体重从小到大排序，将排序结果存放到数组 sortedIndex 里面*/
public static void sortByWeights(Student[] students, int[] sortedIndex)
{
    if (students[0].getWeight() == 0f)//判断是否是非法操作
    {
        System.out.println("非法操作，请输入学生信息之后再进行此项操作");
    }
    float[] weights = getWeight(students);
    float minValue = 1000f;// 记录最小值
    int minSub = 0;// 记录最小值的下标
    int k = 0;// 用于 sortedIndex 下标自增
    for (int j = 0; j < weights.length; j++)
    {
```

```
                for (int i = 0; i < weights.length; i++)
                {
                    if (weights[i]>0)//判断参与比较的数据是否是合法的，如果不合法，就剔除掉，不参与
比较
                    {
                        if (weights[i] <= minValue)//在数组里面找最小值，并且记录下标
                        {
                            minValue = weights[i];//迭代最小值
                            minSub = i;//记录当前最小值下标
                        }
                    }
                }
                weights[minSub] = -1f;//将数组里面最小的值剔除，不再参与比较
                minValue = 1000f;//重设最小值
                sortedIndex[k] = minSub;//将最小值下标按顺序记录在数组 sortedIndex 中
                k++;
            }
        }
}

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(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
    }

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

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

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

/**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;
    }

}
```

程序运行截图：

首先输入学生个数：

```
请输入学生数目:10
```

菜单：

```
本程序功能：
1.随机生成学生信息
2.打印学生信息
3.按学号进行小到大排序
4.按姓名进行小到大排序
5.按身高进行小到大排序
6.按体重进行小到大排序
7.按BMI进行小到大排序
8.查看学生BMI统计信息
9.退出程序
```

功能一：随机生成学生信息

```
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201001    Ewwoe     1.75      64.04     20.98     Normal Range
201002    Wadfz     1.64      58.15     21.70     Normal Range
201003    Rquar     1.62      63.53     24.30     Overweight-At Risk
201004    Kzdcx     1.91      73.17     20.03     Normal Range
201005    Zdjca     1.83      51.10     15.33     Underweight
201006    Werqs     1.61      87.49     33.55     Overweight-Severely Obese
201007    Elfrc     1.99      99.95     25.21     Overweight-Moderately Obese
201008    Hrcfn     1.84      45.25     13.41     Underweight
201009    Jnago     1.69      49.54     17.29     Underweight
201010    Ytcyf     1.88      67.13     18.92     Normal Range
```

功能二：打印学生信息

```
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201001    Ewwoe     1.75      64.04     20.98     Normal Range
201002    Wadfz     1.64      58.15     21.70     Normal Range
201003    Rquar     1.62      63.53     24.30     Overweight-At Risk
201004    Kzdcx     1.91      73.17     20.03     Normal Range
201005    Zdjca     1.83      51.10     15.33     Underweight
201006    Werqs     1.61      87.49     33.55     Overweight-Severely Obese
201007    Elfrc     1.99      99.95     25.21     Overweight-Moderately Obese
201008    Hrcfn     1.84      45.25     13.41     Underweight
201009    Jnago     1.69      49.54     17.29     Underweight
201010    Ytcyf     1.88      67.13     18.92     Normal Range
```

功能三：按学号从小到大排序

```
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201001    Ewwoe     1.75      64.04     20.98     Normal Range
201002    Wadfz     1.64      58.15     21.70     Normal Range
201003    Rquar     1.62      63.53     24.30     Overweight-At Risk
201004    Kzdcx     1.91      73.17     20.03     Normal Range
201005    Zdjca     1.83      51.10     15.33     Underweight
201006    Werqs     1.61      87.49     33.55     Overweight-Severely Obese
201007    Elfrc     1.99      99.95     25.21     Overweight-Moderately Obese
201008    Hrcfn     1.84      45.25     13.41     Underweight
201009    Jnago     1.69      49.54     17.29     Underweight
201010    Ytcyf     1.88      67.13     18.92     Normal Range
```

功能四：按姓名进行小到大排序

```
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201007   Elfrc    1.99      99.95     25.21     Overweight-Moderately Obese
201001   Ewwoe    1.75      64.04     20.98     Normal Range
201008   Hrcfn    1.84      45.25     13.41     Underweight
201009   Jnago    1.69      49.54     17.29     Underweight
201004   Kzdcx    1.91      73.17     20.03     Normal Range
201003   Rquar    1.62      63.53     24.30     Overweight-At Risk
201002   Wadfz    1.64      58.15     21.70     Normal Range
201006   Werqs    1.61      87.49     33.55     Overweight-Severely Obese
201010   Ytcyf    1.88      67.13     18.92     Normal Range
201005   Zdjca    1.83      51.10     15.33     Underweight
```

功能五：按身高进行小到大排序

```
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201006   Werqs    1.61      87.49     33.55     Overweight-Severely Obese
201003   Rquar    1.62      63.53     24.30     Overweight-At Risk
201002   Wadfz    1.64      58.15     21.70     Normal Range
201009   Jnago    1.69      49.54     17.29     Underweight
201001   Ewwoe    1.75      64.04     20.98     Normal Range
201005   Zdjca    1.83      51.10     15.33     Underweight
201008   Hrcfn    1.84      45.25     13.41     Underweight
201010   Ytcyf    1.88      67.13     18.92     Normal Range
201004   Kzdcx    1.91      73.17     20.03     Normal Range
201007   Elfrc    1.99      99.95     25.21     Overweight-Moderately Obese
```

功能六：按体重进行小到大排序

```
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201008   Hrcfn    1.84      45.25     13.41     Underweight
201009   Jnago    1.69      49.54     17.29     Underweight
201005   Zdjca    1.83      51.10     15.33     Underweight
201002   Wadfz    1.64      58.15     21.70     Normal Range
201003   Rquar    1.62      63.53     24.30     Overweight-At Risk
201001   Ewwoe    1.75      64.04     20.98     Normal Range
201010   Ytcyf    1.88      67.13     18.92     Normal Range
201004   Kzdcx    1.91      73.17     20.03     Normal Range
201006   Werqs    1.61      87.49     33.55     Overweight-Severely Obese
201007   Elfrc    1.99      99.95     25.21     Overweight-Moderately Obese
```

功能七：按 BMI 进行小到大排序

```
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201008   Hrcfn    1.84      45.25     13.41     Underweight
201005   Zdjca    1.83      51.10     15.33     Underweight
201009   Jnago    1.69      49.54     17.29     Underweight
201010   Ytcyf    1.88      67.13     18.92     Normal Range
201004   Kzdcx    1.91      73.17     20.03     Normal Range
201001   Ewwoe    1.75      64.04     20.98     Normal Range
201002   Wadfz    1.64      58.15     21.70     Normal Range
201003   Rquar    1.62      63.53     24.30     Overweight-At Risk
201007   Elfrc    1.99      99.95     25.21     Overweight-Moderately Obese
201006   Werqs    1.61      87.49     33.55     Overweight-Severely Obese
```

功能八：查看 BMI 统计信息（分别为查看学生 BMI 的均值，中值，众数，方差）

1. 查看学生 BMI 的均值

```
请选择:1
学生BMI的平均值为:22.30
```

2. 查看学生 BMI 的中值

```
请选择:2
学生BMI的中位数为:33.74
```

3. 查看学生 BMI 的众数

```
BMI众数统计情况:13.52      19.10   21.88   26.08   26.22   33.49   33.98   35.33   37.88   41.81
每个数值出现次数:    1          1       1       1       1       1       1       1       1       1
众数不存在!
```

4. 查看学生 BMI 的方差

```
请选择:4
学生的BMI的方差为:73.24
```

功能九：退出程序

```
9
It is the end!
```