Java 程序设计实验报告

学号：　1170300725

姓名：　周孟伦

专业：　计算机科学与技术

班级：　　1703007

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

```
package edu.hit.java.exp2.hit1170300725;
import java.math.BigDecimal;
import java.util.*;

class Student{
    private String id;
    private String name;
    private float height;
    private float weight;
    private float BMI;
    private String health;

    public void setId(String id){
        this.id = id;
    }

    public String getId(){
        return id;
    }

    public void setName(String name){
        this.name = name;
    }

    public String getName(){
        return name;
    }

    public void setHeight(float height){
        this.height = height;
    }

    public float getHeight(){
        return height;
    }

    public void setWeight(float weight){
        this.weight = weight;
    }

    public float getWeight(){
        return weight;
    }

    public void setBMI(float BMI){
        this.BMI = BMI;
    }

    public float getBMI(){
        return BMI;
    }

    public void setHealth(String health){
        this.health = health;
    }

    public String getHealth(){
```

```java
        return health;
    }

    public String toString(){
        return getId() + "\t" + getName() + "\t" +
                String.format("%.2f", getHeight()) + "\t" +
                String.format("%.2f", getWeight()) + "\t" +
                String.format("%.2f", getBMI()) + "\t" + getHealth();
    }
}
public class OOBMI {
       private static Random random;
       /*
        * define universe random
        */
/**
 *
 * @param args
 */
       public static void main(String[] args) {

            random = new Random();
            /*
             * creat new rondom
             */
            Student[] students=null;
            /*
             * shouid not be undefined
             */
               menu(students);

 }
/**
 *
 * @return
 */
 public static int printmenu() {
     Scanner sc=new Scanner(System.in);
     System.out.println("1:generrate student's information");
     System.out.println("2:print stuent's information");
     System.out.println("3:sort by ID");
     System.out.println("4:sort by height");
     System.out.println("5:sort by weight");
     System.out.println("6:sort by name");
    System.out.println("7:sort by bmi");
    System.out.println("8.print statics");
    System.out.println("9:exit");
    int choice=sc.nextInt();
    return choice;
}
/*
 * get choice
 */
/**  函数名: menu
 * function:
 * 1.choose which to function
 * 2. get the student number
 * 3.exit the program
 * @param ids
 * @param names
 * @param heights
 * @param weights
 * @param bmis
 */
public static void menu(Student[] students) {
    Scanner sc=new Scanner(System.in);
    int[] sortedIndex=null;
    boolean quit=false;
    /**
     * if quit = false,continue
     * else if quit = true,quit the program end
     */
    while(!quit) {
        int choice=printmenu();
        if(choice==1) {
            System.out.println("please input student's number");
           int num=sc.nextInt();
           students=new Student[num];
           for(int i=0;i<num;i++) {
               students[i]=new Student();
           }
           sortedIndex=genStudents(num,students);
           /*
            * random students' data
            */
        }else if(choice==2) {
            printStudents(sortedIndex,students);
        }else if(choice==3) {
             sortedIndex=sortByID(students);
         }else if(choice==4) {
             sortedIndex=sortByHeight(students);
         }else if(choice==5) {
             sortedIndex=sortByWeight(students);
         }else if(choice==6) {
             sortedIndex=sortByName(students);
         }else if(choice==7) {
             sortedIndex=sortByBmi(students);
         }else if(choice==8) {
             printStatics(sortedIndex,students);
         }
         else if(choice==9) {
             quit=true;
             /*
              * quit the program
              */
         }
     }
 }
 /**
  *
  * @param num
  * @param students
  * @return
  */
 private static int[] genStudents(int num,Student[] students) {
     int[] index=new int[num];
     for(int i = 0; i < num; i ++){
         String tempId = String.valueOf(random.nextInt(1000) + 1000);
         while(isExists(num,tempId,students)){
             tempId = String.valueOf(random.nextInt(1000) + 1000);
         }
         students[i].setId(tempId);
         String chars = "abcdefghijklmnopqrstuvwxyz";
         students[i].setName(String.valueOf(chars.charAt(random.nextInt(26)))
+ chars.charAt(random.nextInt(26)) + chars.charAt(random.nextInt(26)) +
chars.charAt(random.nextInt(26)));
         students[i].setHeight(random.nextFloat() * 50f + 150f);
         //height ranges from 150 to 200
         students[i].setWeight(random.nextFloat() * 25f + 50);
         //weight range from 50 to 74

students[i].setBMI(students[i].getWeight()*10000/students[i].getHeight()/st
udents[i].getHeight());
         //calculate the bmi using the given height and weight
         index[i]=i;
         students[i].setHealth(checkHealth(students[i].getBMI()));
         //check the health and store
     }
     return index;
 }
 /**
  *
  * @param mt
  * @return
  */
 private static float round(float mt){
     BigDecimal bmt = new BigDecimal(mt);
     float nmt =
             bmt.~~setScale~~(2, BigDecimal.~~ROUND_HALF_UP~~).floatValue();
     return nmt;
 }
 /**
  *
  * @param num
  * @param id
  * @param students
  * @return
  */
 private static boolean isExists(int num,String id,Student[] students){
     for(int i = 0; i < num; i ++){
         if(students[i].getId() == null){
             break;
         }else if(students[i].getId().equals(id)){
             return false;
         }
     }
     return false;
 }
 /**
  *
  * @param sorted
  * @param students
  */
 private static void printStatics(int[] sorted,Student[] students){
     printStudents(sorted,students);
     //System.out.println();
```

```
int number=students.length;
float[] bmis = new float[number];
float sum = 0;
float average;
/*
    calculate the sum of bmi
 */
for(int i = 0; i < number; i ++){
    bmis[i] = students[i].getBMI();
    sum += bmis[i];
}
average = round(sum / number);
Arrays.sort(bmis);  //sort the bmi
float midNum;
/*
    get the mid number
 */
if(number % 2 == 0){
    midNum = (bmis[number / 2] + bmis[number / 2 + 1]) / 2;
}else{
    midNum = bmis[number / 2 + 1];
}
/*
    using the HashSet because it cannot store the duplicated data
 */
HashSet<Float> uniqueData = new HashSet<>();
HashMap<Integer, Float> mass = new HashMap<>();
for(int i = 0; i < number; i ++){
    uniqueData.add(bmis[i]);
}
int[] count = new int[uniqueData.size()];
int j = 0;
/*
    calculate the frequency of every number
 */
for(float f1 : uniqueData){
    for(float f2 : bmis){
        if(f1 == f2){
            count[j]++;
        }
    }
    mass.put(count[j], f1);
    j ++;
}
```

```
        int k = 0;
        for(int i : count){
            k = Math.max(k, i);
        }
        float massNum = mass.get(k);
        float variance;
        float tempSum = 0;
        for(int i = 0; i < number; i ++){
            tempSum += Math.pow(bmis[i] - average, 2);
        }
        variance = tempSum / number;
        System.out.println("Average: " + String.format("%.2f",average));
        System.out.println("Mediant: " + String.format("%.2f",midNum));
        System.out.println("Mode: " + String.format("%.2f",massNum));
        System.out.println("Variance: " + String.format("%.2f",variance));
    }

    private static void printStudents(int[] sortedindex,Student[] students){
        for(int tempInt : sortedindex){
            System.out.println(students[tempInt].toString());
        }
    }
    /**函数名: sortByBmi
     * funcition:
     * 1.sort students by bmi
     * 2.return the array
     * @param bmis
     * @return
     */
    public static int[] sortByBmi(Student[] students) {
        int num=students.length;
        int[] sortedIndex=new int[num];
        int tmp=0;
        for(int i=0;i<num;i++)
            sortedIndex[i]=i;
        float[] tempArray = new float[num];
        for(int i = 0; i < num; i ++){
            tempArray[i] = students[i].getBMI();
        }
        for(int x=0;x<num;x++) {
            for(int y=x+1;y<num;y++) {
                if(tempArray[x]>tempArray[y]) {
                    Float tempHeight = tempArray[y];
                   tempArray[y] = tempArray[x];
```

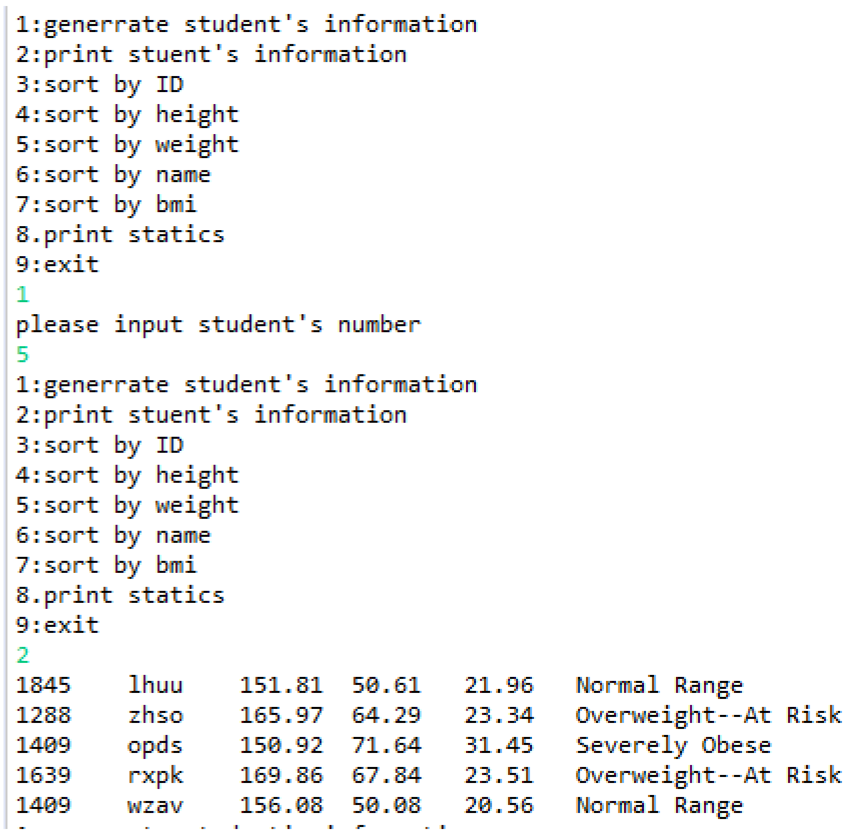
```
                    tempArray[x] = tempHeight;
                     tmp=sortedIndex[y];
                     sortedIndex[y]=sortedIndex[x];
                     sortedIndex[x]=tmp;
                }
            }
        }
           return sortedIndex;
    }
    /**函数名: sortByNmae
     * funcition:
     * 1.sort students by name
     * 2.return the array
     * @param bmis
     * @return
     */
    public static int[] sortByName(Student[] students) {
        int num=students.length;
        int[] sortedIndex=new int[num];
        int tmp=0;
        for(int i=0;i<num;i++)
            sortedIndex[i]=i;
        String[] tempArray = new String[num];
        for(int i = 0; i < num; i ++){
            tempArray[i] = students[i].getName();
        }
        for(int x=0;x<num;x++) {
            for(int y=x+1;y<num;y++) {
                if(tempArray[x].compareTo(tempArray[y]) > 0) {
                    String tempHeight = tempArray[y];
                    tempArray[y] = tempArray[x];
                    tempArray[x] = tempHeight;
                     tmp=sortedIndex[y];
                     sortedIndex[y]=sortedIndex[x];
                     sortedIndex[x]=tmp;
                }
            }
        }
           return sortedIndex;
    }
    /**函数名: sortByweight
     * funcition:
     * 1.sort students by weight
     * 2.return the array
```

```
 * @param weights
 * @return
 */
public static int[] sortByWeight(Student[] students) {
    int num=students.length;
    int[] sortedIndex=new int[num];
    int tmp=0;
    for(int i=0;i<num;i++)
        sortedIndex[i]=i;
    float[] tempArray = new float[num];
    for(int i = 0; i < num; i ++){
        tempArray[i] = students[i].getWeight();
    }
    for(int x=0;x<num;x++) {
        for(int y=x+1;y<num;y++) {
            if(tempArray[x]>tempArray[y]) {
                Float tempHeight = tempArray[y];
               tempArray[y] = tempArray[x];
               tempArray[x] = tempHeight;
                tmp=sortedIndex[y];
                sortedIndex[y]=sortedIndex[x];
                sortedIndex[x]=tmp;
            }
        }
    }
       return sortedIndex;
}
/**函数名: sortByheight
 * funcition:
 * 1.sort students by height
 * 2.return the array
 * @param heights
 * @return
 */
public static int[] sortByHeight(Student[] students) {
    int num=students.length;
    int[] sortedIndex=new int[num];
    int tmp=0;
    for(int i=0;i<num;i++)
        sortedIndex[i]=i;
    float[] tempArray = new float[num];
    for(int i = 0; i < num; i ++){
        tempArray[i] = students[i].getHeight();
    }
    for(int x=0;x<num;x++) {
        for(int y=x+1;y<num;y++) {
            if(tempArray[x]>tempArray[y]) {
                Float tempHeight = tempArray[y];
               tempArray[y] = tempArray[x];
               tempArray[x] = tempHeight;
                tmp=sortedIndex[y];
                sortedIndex[y]=sortedIndex[x];
                sortedIndex[x]=tmp;
            }
        }
    }
       return sortedIndex;
}
/**函数名: sortByID
 * funcition:
 * 1.sort students by ID
 * 2.return the array
 * @param ids
 * @return
 */
public static int[] sortByID(Student[] students) {
    int num=students.length;
    int[] sortedIndex=new int[num];
    int tmp=0;
    for(int i=0;i<num;i++)
        sortedIndex[i]=i;
    String[] tempArray = new String[num];
    for(int i = 0; i < num; i ++){
        tempArray[i] = students[i].getId();
    }
    for(int x=0;x<num;x++) {
        for(int y=x+1;y<num;y++) {
            if(tempArray[x].compareTo(tempArray[y]) > 0) {
                String tempHeight = tempArray[y];
               tempArray[y] = tempArray[x];
               tempArray[x] = tempHeight;
                tmp=sortedIndex[y];
                sortedIndex[y]=sortedIndex[x];
                sortedIndex[x]=tmp;
            }
        }
    }
       return sortedIndex;
 }
 /**函数名: inputstudents
  * function
  * 1.get the information
  * 2.caculate the bmi
  * 3.show the healthy condition
  * @param num
  * @param ids
  * @param names
  * @param heights
  * @param weights
  * @param bmis
  * @return
  */

 /**函数名:  checkhealth
  * function
  * 1.get bmi
  * 2.choice the healthy condition
  * 3.renturn result
  * @param bmi
  * @return
  */
 public static String checkHealth(float bmi) {
      if (bmi <= 18.5) {
          return "Underweight";
      } else if (bmi <= 23) {
          return "Normal Range";
      } else if (bmi <= 25) {
          return "Overweight--At Risk";
      } else if (bmi <= 30) {
          return "Overweight--Moderately Obese";
      } else {
          return "Severely Obese";
      }
}
}
```

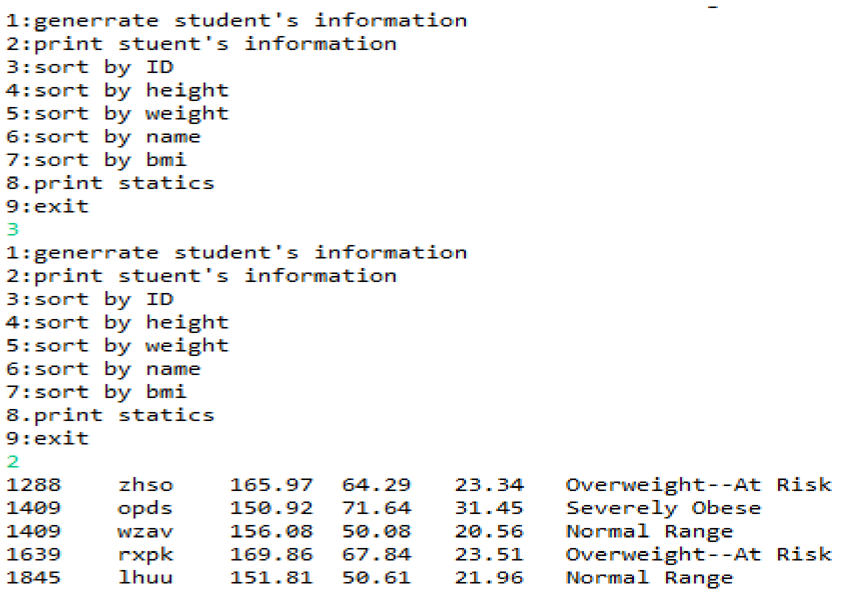
截图展示:

1.随机生成学生信息并打印



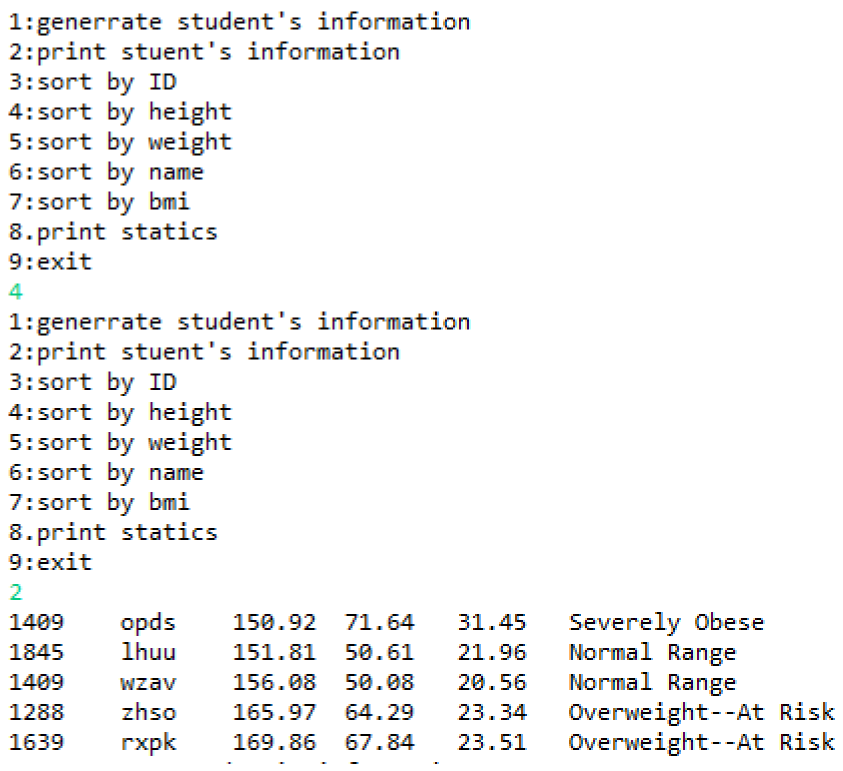
```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
1
please input student's number
5
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
2
1845    lhuu    151.81  50.61   21.96   Normal Range
1288    zhso    165.97  64.29   23.34   Overweight--At Risk
1409    opds    150.92  71.64   31.45   Severely Obese
1639    rxpk    169.86  67.84   23.51   Overweight--At Risk
1409    wzav    156.08  50.08   20.56   Normal Range
```

2.按学号排序并打印



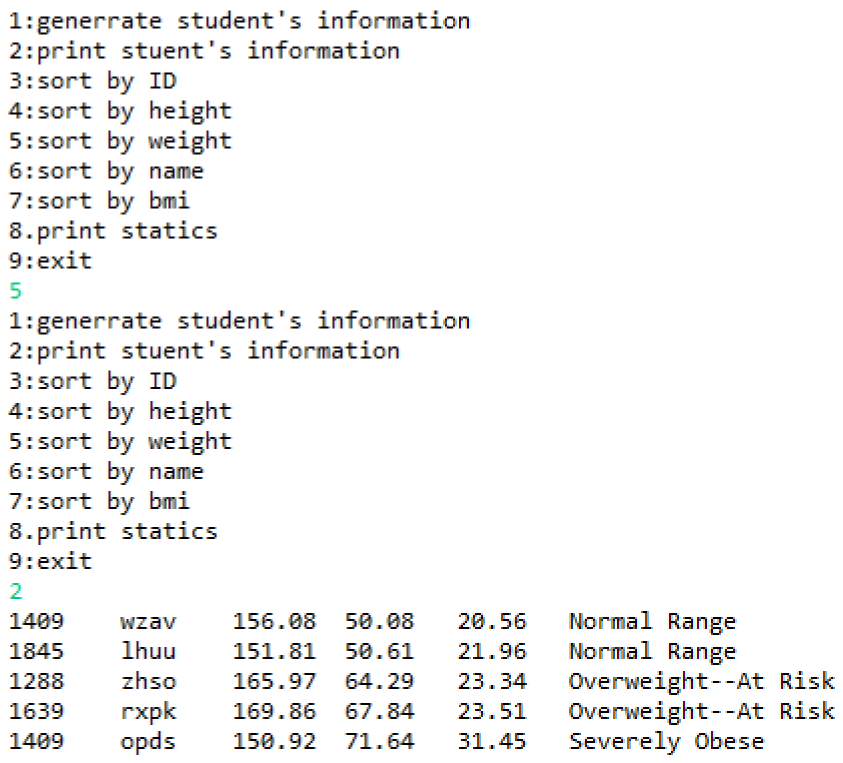
```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
3
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
2
1288    zhso    165.97  64.29   23.34   Overweight--At Risk
1409    opds    150.92  71.64   31.45   Severely Obese
1409    wzav    156.08  50.08   20.56   Normal Range
1639    rxpk    169.86  67.84   23.51   Overweight--At Risk
1845    lhuu    151.81  50.61   21.96   Normal Range
```

3.按身高排序并打印



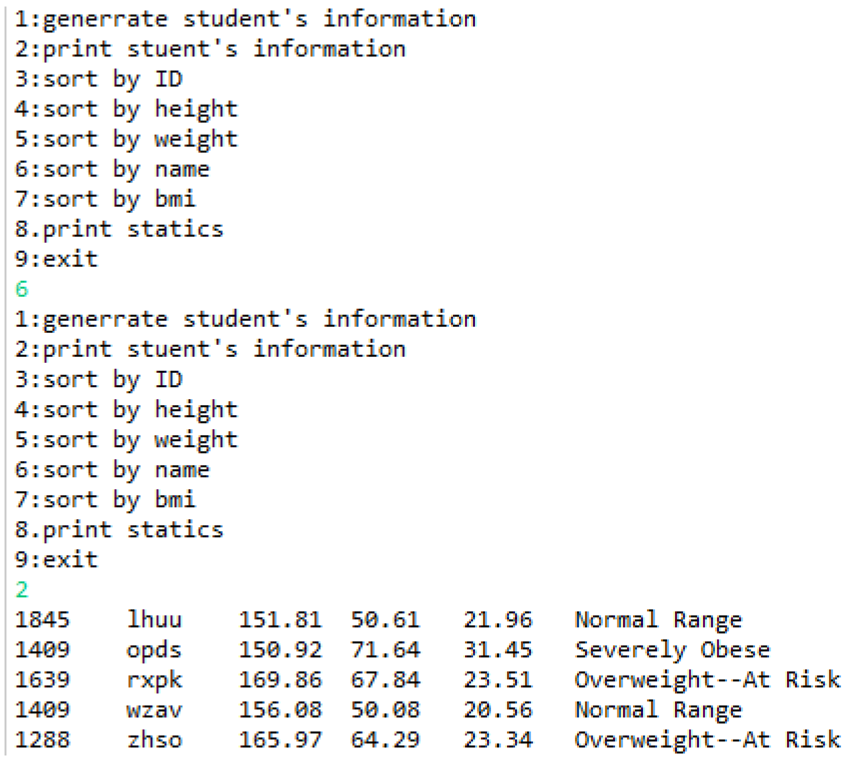
```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
4
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
2
1409    opds    150.92  71.64   31.45   Severely Obese
1845    lhuu    151.81  50.61   21.96   Normal Range
1409    wzav    156.08  50.08   20.56   Normal Range
1288    zhso    165.97  64.29   23.34   Overweight--At Risk
1639    rxpk    169.86  67.84   23.51   Overweight--At Risk
```

4.按体重排序并打印



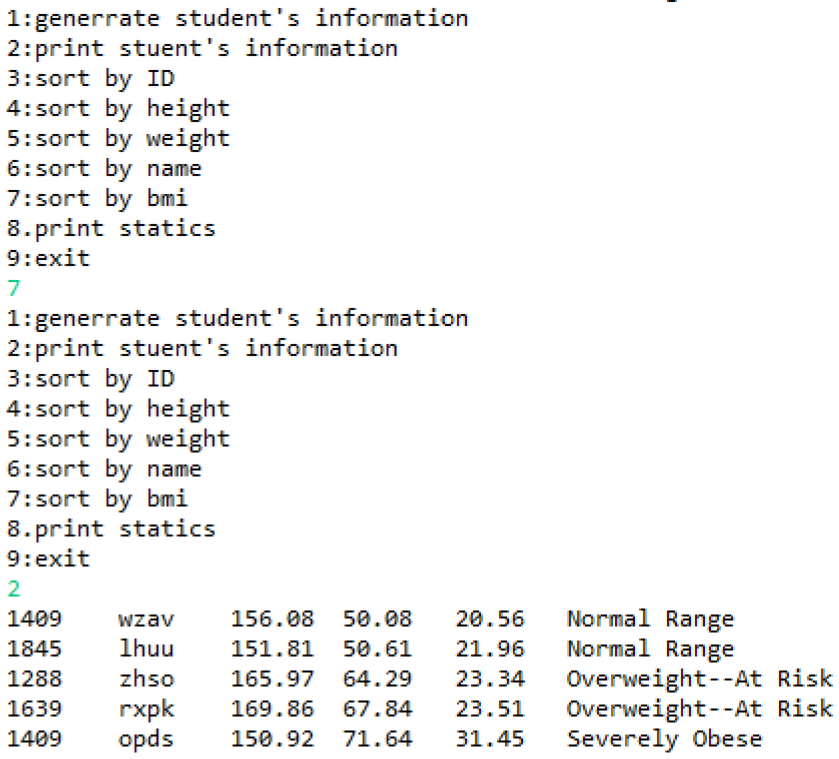
```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
5
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
2
1409    wzav    156.08  50.08   20.56   Normal Range
1845    lhuu    151.81  50.61   21.96   Normal Range
1288    zhso    165.97  64.29   23.34   Overweight--At Risk
1639    rxpk    169.86  67.84   23.51   Overweight--At Risk
1409    opds    150.92  71.64   31.45   Severely Obese
```

5.按姓名排序并打印



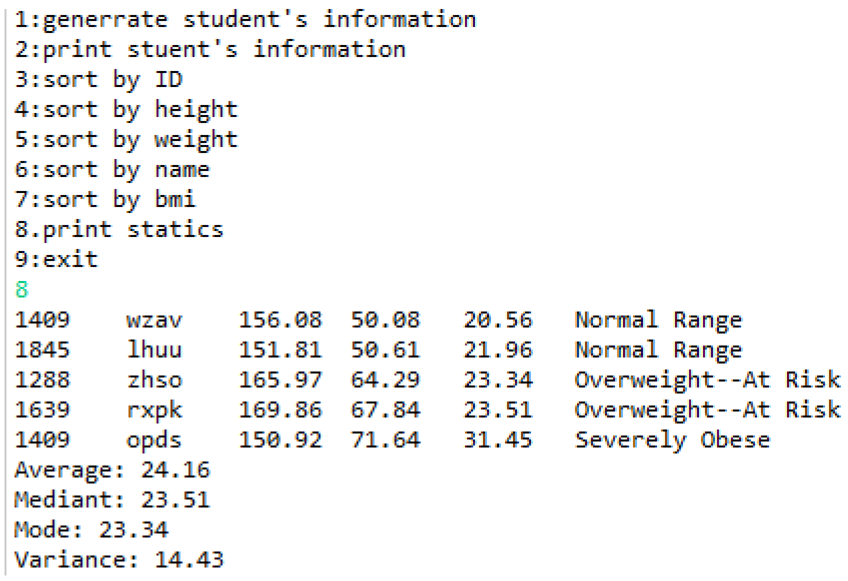
```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
6
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
2
1845    lhuu    151.81  50.61   21.96   Normal Range
1409    opds    150.92  71.64   31.45   Severely Obese
1639    rxpk    169.86  67.84   23.51   Overweight--At Risk
1409    wzav    156.08  50.08   20.56   Normal Range
1288    zhso    165.97  64.29   23.34   Overweight--At Risk
```

6.按 BMI 值排序并打印



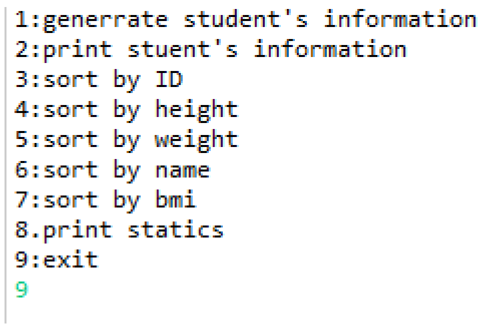
```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
7
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
2
1409    wzav    156.08  50.08   20.56   Normal Range
1845    lhuu    151.81  50.61   21.96   Normal Range
1288    zhso    165.97  64.29   23.34   Overweight--At Risk
1639    rxpk    169.86  67.84   23.51   Overweight--At Risk
1409    opds    150.92  71.64   31.45   Severely Obese
```

7.打印所有学生数据及 BMI 中位数，众数，平均数，方差



```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
8
1409    wzav    156.08  50.08   20.56   Normal Range
1845    lhuu    151.81  50.61   21.96   Normal Range
1288    zhso    165.97  64.29   23.34   Overweight--At Risk
1639    rxpk    169.86  67.84   23.51   Overweight--At Risk
1409    opds    150.92  71.64   31.45   Severely Obese
Average: 24.16
Mediant: 23.51
Mode: 23.34
Variance: 14.43
```

8.退出程序



```
1:generrate student's information
2:print stuent's information
3:sort by ID
4:sort by height
5:sort by weight
6:sort by name
7:sort by bmi
8.print statics
9:exit
9
```