实验一：Java 基本程序设计

一、实验目的

1）掌握标准输入输出函数的使用。
2）静态函数的定义和使用（本实验要求所有函数均为静态函数）；
2）掌握简单排序算法；
3）Java 基础语法综合运行（非面向对象版本 BMI 程序）；

二、实验内容

1)编写 BMI 类，在 main 函数中增加数组 String[] ids，String[] names，float[] heights, float[] weights, float[] bmis，分别存储学生们的学号、姓名、身高、体重、计算后的 bmi 值和胖瘦健康状况。注意，上述数值均需保留两位小数存储。

2）定义 inputStudents 函数，该函数的参数为上述数组，该函数的功能是输入多个学生的相关信息，并将相关数据存储到上述数组中；

3）在 BMI 类中，增加一个函数 checkHealth，函数参数为 bmi 值，该函数按下表中 BMI 取值范围判断胖瘦健康状况，该函数的返回值为字符串，返回结果即下表中的第一列中的值，并在 inputStudents 函数中调用该函数，获得学生的胖瘦健康状况。

Category	BMI (kg/m^2)	
	from	to
Underweight		18.5
Normal Range	18.5	23
Overweight—At Risk	23	25
Overweight—Moderately Obese	25	30
Overweight—Severely Obese	30	

4）在 BMI 类中，增加 5 个排序 sortByXXX 函数，XXX 表示排序属性，可以分别按照学生学号、姓名、身高、体重、BMI 进行由小到大排序，排序算法可以利用简单排序、选择排序、冒泡排序算法或其他算法（选择其中一种算法实现即可）。排序前后必须保证同一个学生在所有数组中对应相同的下标！**为了方便实现上述功能,可定义一个排序数组 int sortedIndex[]，该数组中保存了进行排序的数组排序后的下标，排序结束后，返回该数组，以便根据该数据进行打印显示。**

5）在 BMI 类中，增加 printStudents 函数，该函数的参数含有 int sortedIndex[],该函数可以打印排序前和排序后的结果。打印时，每个学生的信息打印为一行，为了清晰，学号、姓名、身高、体重和计算后的 bmi 值之间用制表符(\t)隔开。

6）定义 menu 函数，提供输入学生、打印学生，5 种排序、程序退出等 8 种选项，用户输入指定选项后，运行相应函数功能。**注意，在调用 inputStudents 函数前，需先提示用户数输入指定人数。**

7）在 BMI 类的 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.exp1.hit1173710105;

import java.util.*;

public class BMI
{
	public static void main(String[] args)
	{
		// TODO Auto-generated method stub
		String[] ids = null;// 存储学生学号
		String[] names = null;// 存储学生姓名
		float[] heights = null;// 存储学生身高
		float[] weights = null;// 存储学生体重
		float[] bmis = null;// 存储学生 bmi 值
		String[] PhysicalCondition = null;//用于记录学生的健康状况
		int[] sortedIndex = null;// 该数组中保存了进行排序的数组排序后的下标
		int i;

		int num = 0;// 确认学生数目
		Scanner scanner = new Scanner(System.in);
		System.out.print("请输入学生数目:");
		num = scanner.nextInt();

		//创建数组，确认数组的长度
		ids = new String[num];
		names = new String[num];
		heights = new float[num];
		weights = new float[num];
		bmis = new float[num];
		sortedIndex = new int[num];
		PhysicalCondition = new String[num];

		do
		{
			i = menu();// 获取用户想执行功能
			switch (i)
```

```
                {
                case 1:
                    //输入学生信息
                    BMI.inputStudents(ids, names, heights, weights, bmis, sortedIndex,
PhysicalCondition,num);
                    break;
                case 2:
                    //打印学生信息
                    BMI.printStudents(ids, names, heights, weights, bmis, sortedIndex,
PhysicalCondition);
                    break;
                case 3:
                    //按学号从小到大排序
                    BMI.sortByID(ids, sortedIndex);
                    break;
                case 4:
                    //按姓名从小到大排序
                    BMI.sortByName(names, sortedIndex);
                    break;
                case 5:
                    //按身高从小到大排序
                    BMI.sortByHeights(heights, sortedIndex);
                    break;
                case 6:
                    //按体重从小到大排序
                    BMI.sortByWeights(weights, sortedIndex);
                    break;
                case 7:
                    //按 BMI 从小到大排序
                    BMI.sortByBMI(bmis, sortedIndex);
                    break;
                case 8:
                    //程序结束，退出
                    System.out.println("It is the end!");
                    scanner.close();
                    System.exit(0);
                    break;
                default:
                    break;
                }
            } while (true);
        }

        /**
```

```
     * 用户通过键盘输入学生信息 所接收到的信息将会存储到数组之中
     */
    public static void inputStudents(String[] ids, String[] names, float[] heights, float[] weights,
float[] bmis,
            int[] sortedIndex, String[] PhysicalCondition,int num)
    {

        Scanner scanner = new Scanner(System.in);
        for (int i = 0; i < num; i++)
        {
            System.out.println("现请输入第"+(i+1)+"名学生的信息:");
            System.out.print("请输入学生学号:");
            ids[i] = scanner.next();
            System.out.print("请输入学生姓名:");
            names[i] = scanner.next();
            System.out.print("请输入学生身高(单位 m):");
            heights[i] = scanner.nextFloat();
            System.out.print("请输入学生体重(单位 kg):");
            weights[i] = scanner.nextFloat();
            bmis[i] = weights[i] / (heights[i] * heights[i]);
            PhysicalCondition[i] = checkHealth(bmis[i]);
        }
        // 默认顺序
        for (int i = 0; i < sortedIndex.length; i++)
        {
            sortedIndex[i] = i;
        }
    }

    /**
     * 按格式，按顺序打印学生信息*/
    public static void printStudents(String[] ids, String[] names, float[] heights, float[] weights,
float[] bmis,
            int[] sortedIndex, String[] PhysicalCondition)
    {
        if (ids == null)
        {
            System.out.println("非法操作，请输入学生信息之后再进行此项操作");
        }
        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
        }
    }

    /**
     * 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
            {
                BMI.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 > 8 || i < 0);
```

```
        return i;
}

/**
 * 打印菜单信息*/
public static void printMenuMessage()
{
        System.out.println("本程序功能：");
        System.out.println("*注:在实现其他功能之前，请先输入学生信息");
        System.out.println("1.输入学生信息");
        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.退出程序");
}

/**
 * 根据每个学生的 BMI 的值判断其健康状况
 * 返回一个代表健康状况的字符标识符*/
public static 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;
```

```
    }

    /**
     * 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(String[] ids, int[] sortedIndex)
    {
        if (ids == 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");
        }
        String[] temp = ids.clone();//将数组复制一份，以免误改原本数组
        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 (temp[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 (temp[i].compareTo(minValue) <= 0)//在数组里面找最小值，并且记录下标
                    {
                        minValue = temp[i];//迭代最小值
                        minSub = i;//记录当前最小值下标
                    }
                }
            }
            temp[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(String[] names, int[] sortedIndex)
    {
        if (names == 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");
        }
```

```
        String[] temp = names.clone();//将数组复制一份，以免误改原本数组
        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 (temp[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 (temp[i].compareTo(minValue) <= 0)//在数组里面找最小值，并且记录下标
                    {
                        minValue = temp[i];//迭代最小值
                        minSub = i;//记录当前最小值下标
                    }
                }
            }
            temp[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(float[] heights, int[] sortedIndex)
    {
        if (heights == 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");
        }
        float[] temp = heights.clone();//将数组复制一份，以免误改原本数组
        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 (temp[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 (temp[i] <= minValue)//在数组里面找最小值，并且记录下标
                    {
                        minValue = temp[i];//迭代最小值
                        minSub = i;//记录当前最小值下标
                    }
                }
            }
            temp[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++;
        }
    }

    public static void sortByBMI(float[] bmis, int[] sortedIndex)
    {
        if (bmis == 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");
        }
        float[] temp = bmis.clone();//将数组复制一份，以免误改原本数组
        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 (temp[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 (temp[i] <= minValue)//在数组里面找最小值，并且记录下标
                    {
                        minValue = temp[i];//迭代最小值
                        minSub = i;//记录当前最小值下标
                    }
                }
            }
            temp[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++;
        }
```

```
    }

    public static void sortByWeights(float[] weights, int[] sortedIndex)
    {
        if (weights == 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");
        }
        float[] temp = weights.clone();//将数组复制一份，以免误改原本数组
        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 (temp[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 (temp[i] <= minValue)//在数组里面找最小值，并且记录下标
                    {
                        minValue = temp[i];//迭代最小值
                        minSub = i;//记录当前最小值下标
                    }
                }
            }
            temp[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++;
        }
    }
}
```

运行结果：

菜单：

```
本程序功能：
*注：在实现其他功能之前，请先输入学生信息
1.输入学生信息
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.退出程序
```

功能一：输入学生信息

```
请输入学生数目:5
本程序功能：
*注：在实现其他功能之前，请先输入学生信息
1.输入学生信息
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.退出程序
1
现请输入第1名学生的信息：
请输入学生学号:201001
请输入学生姓名:Jack
请输入学生身高(单位m):1.8
请输入学生体重(单位kg):90
现请输入第2名学生的信息：
请输入学生学号:201002
请输入学生姓名:Marry
请输入学生身高(单位m):1.7
请输入学生体重(单位kg):55
```

```
现请输入第3名学生的信息：
请输入学生学号:201003
请输入学生姓名:Nikolas
请输入学生身高(单位m):1.6
请输入学生体重(单位kg):80
现请输入第4名学生的信息：
请输入学生学号:201004
请输入学生姓名:Tom
请输入学生身高(单位m):1.8
请输入学生体重(单位kg):55
现请输入第5名学生的信息：
请输入学生学号:201005
请输入学生姓名:Neymar
请输入学生身高(单位m):1.76
请输入学生体重(单位kg):60
```

功能二：按顺序打印学生信息

```
2
学生信息总览：
学号      姓名      身高      体重      BMI指数     健康状况
201001   Jack     1.80     90.00    27.78    Overweight-Moderately Obese
201002   Marry    1.70     55.00    19.03    Normal Range
201003   Nikolas  1.60     80.00    31.25    Overweight-Severely Obese
201004   Tom      1.80     55.00    16.98    Underweight
201005   Neymar   1.76     60.00    19.37    Normal Range
```

功能三：按学号从小到大排序，顺序打印

```
2
学生信息总览:
学号      姓名      身高      体重      BMI指数     健康状况
201001    Jack      1.80      90.00     27.78     Overweight-Moderately Obese
201002    Marry     1.70      55.00     19.03     Normal Range
201003    Nikolas   1.60      80.00     31.25     Overweight-Severely Obese
201004    Tom       1.80      55.00     16.98     Underweight
201005    Neymar    1.76      60.00     19.37     Normal Range
```

功能四：按姓名从小到大排序，然后打印

```
学生信息总览:
学号      姓名      身高      体重      BMI指数     健康状况
201001    Jack      1.80      90.00     27.78     Overweight-Moderately Obese
201002    Marry     1.70      55.00     19.03     Normal Range
201005    Neymar    1.76      60.00     19.37     Normal Range
201003    Nikolas   1.60      80.00     31.25     Overweight-Severely Obese
201004    Tom       1.80      55.00     16.98     Underweight
```

功能五：按身高从小到大排序，然后打印

```
学生信息总览:
学号      姓名      身高      体重      BMI指数     健康状况
201003    Nikolas   1.60      80.00     31.25     Overweight-Severely Obese
201002    Marry     1.70      55.00     19.03     Normal Range
201005    Neymar    1.76      60.00     19.37     Normal Range
201004    Tom       1.80      55.00     16.98     Underweight
201001    Jack      1.80      90.00     27.78     Overweight-Moderately Obese
```

功能六：按体重从小到大排序，然后打印

```
学生信息总览:
学号      姓名      身高      体重      BMI指数     健康状况
201004    Tom       1.80      55.00     16.98     Underweight
201002    Marry     1.70      55.00     19.03     Normal Range
201005    Neymar    1.76      60.00     19.37     Normal Range
201003    Nikolas   1.60      80.00     31.25     Overweight-Severely Obese
201001    Jack      1.80      90.00     27.78     Overweight-Moderately Obese
```

功能七：按 BMI 从小到大排序，然后打印

```
学生信息总览:
学号      姓名      身高      体重      BMI指数     健康状况
201004    Tom       1.80      55.00     16.98     Underweight
201002    Marry     1.70      55.00     19.03     Normal Range
201005    Neymar    1.76      60.00     19.37     Normal Range
201001    Jack      1.80      90.00     27.78     Overweight-Moderately Obese
201003    Nikolas   1.60      80.00     31.25     Overweight-Severely Obese
```

功能八：退出程序

```
8.退出程序
8
It is the end!
```