



HARBIN INSTITUTE OF TECHNOLOGY

哈尔滨工业大学

Java 程序设计实验报告

学号：

姓名：

专业： 软件工程类

班级： 1637102

实验二：Java 基本程序设计

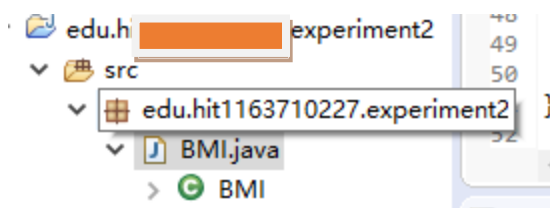
一、实验目的

- 1) 掌握标准输入输出函数的使用。
- 2) 掌握自定义 static 方法及其调用
- 3) 掌握 if 和 if-else 和 switch 分支语句
- 4) 掌握 while、do-while 和 for 循环语句
- 5) 掌握 break 和 continue 程序控制语句
- 6) 掌握嵌套循环的使用方法
- 7) 掌握数组的定义和使用方法
- 8) 了解二维数组的特性
- 9) 掌握简单排序算法
- 10) Java 基础知识综合运行

二、实验内容

1) 编写 BMI 类，定义一个函数 input，该函数实现从键盘输入一个学生的学号、姓名、身高(米)、体重(公斤)，计算 BMI 并打印输出，输出格式为：学号：XXX，姓名：YYY，身高：ZZZ 厘米，体重：MMM 斤，BMI：NNN。

Step1, 创建 BMI 类



Step2, 编写代码。

```
static void Input(){  
    Scanner input = new Scanner(System.in);  
  
    System.out.println("请输入学号: ");  
    long numble=input.nextLong();
```

```
System.out.println("请输入姓名: ");
String name=input.next();

System.out.println("请输入身高(m): ");
double height=input.nextFloat();
System.out.println("请输入体重(Kg): ");
double weight=input.nextFloat();

double bmi=weight/(height*height);

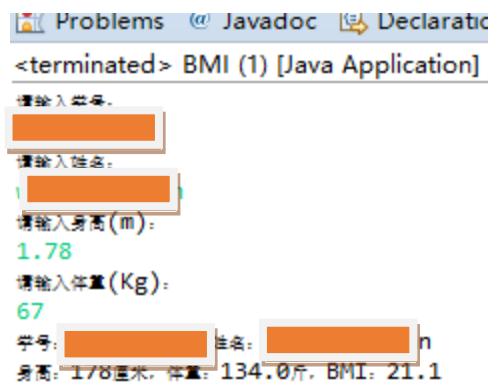
System.out.println("学号: "+numble+"姓名: "+name);
System.out.printf("身高: %.0f"+"厘米, 体重: %.1f"+"斤, BMI: %.1f",height*100,weight*2,bmi);

input.close();
}
```

Step3, 在主函数中调用并观察结果

```
27
28 public static void main(String[] argv){
29     Input();
30 }
31
32
```

结果如下:



Problems @ Javadoc Declaratio

<terminated> BMI (1) [Java Application]

请输入姓名:

请输入身高(m):

1.78

请输入体重(Kg):

67

学号: 姓名: n

身高: 178厘米, 体重: 134.0斤, BMI: 21.1

2) 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 input 函数中调用该函数, 并打印输出学生的胖瘦健康状况。

Step1: 添加 checkHealth 函数:

```

public static String checkHealth(float bmi){
    String str[]={ "Underweight", "Normal Range",
        "Overweight-At Risk", "Overweight-Moderately Obese",
        "Overweight-Severely Obese"};
    if(bmi<18.5)return str[0];
    else if(bmi<23)return str[1];
    else if(bmi<25)return str[2];
    else if(bmi<30)return str[3];
    else return str[4];
}

```

Step2: 修改 input 函数:

```

static void Input(){

    Scanner input = new Scanner(System.in);

    System.out.println("请输入学号: ");
    long numble=input.nextLong();

    System.out.println("请输入姓名: ");
    String name=input.next();

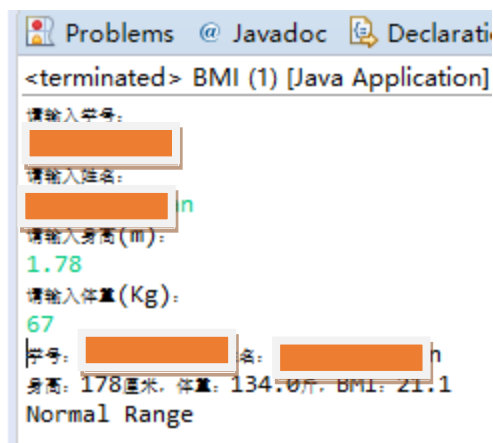
    System.out.println("请输入身高(m): ");
    double height=input.nextFloat();
    System.out.println("请输入体重(Kg): ");
    double weight=input.nextFloat();

    double bmi=weight/(height*height);

    System.out.println("学号: "+numble+"姓名: "+n
    System.out.printf("身高: %.0f"+"厘米, 体重: %.1
    System.out.println(checkHealth(bmi));
    input.close();
}

```

运行结果如下:



3) 改编 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、体重和计算后的 bmi 值；先询问学生的人数，输入指定人数后，依次输入各个学生的学号、姓名、身高、体重，并计算 bmi 值，并将相关数据存储在数组中；

Step1: 依题意添加代码如下：

```
public static void main(String[] agrv){
    Scanner input=new Scanner(System.in);

    System.out.println("请输入学生人数N: ");
    int N=input.nextInt();

    String[] ids=new String[N];
    String[] names=new String[N];
    float[] heights=new float[N];
    float[] weights=new float[N];
    float[] bmis=new float[N];

    System.out.println("请输入每个学生的学号、姓名、身高、体重: ");
    for(int i=0;i<N;i++){
        ids[i]=input.next();
        names[i]=input.next();
        heights[i]=input.nextFloat();
        weights[i]=input.nextFloat();
        bmis[i]=weights[i]/(heights[i]*heights[i]);
    }

    input.close();
}
```

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。

Step1.根据题意增加 SortByXXX 函数如下：

```
1、 public static int[] SortByName(String name[],int N)
```

```
public static int[] SortByName(String name[],int N){
    int sortindex[]=new int[N];
    for(int i=0;i<N;i++)sortindex[i]=i;

    for(int i=0;i<N;i++){
        int min=i;
        for(int j=i+1;j<N;j++){
            if(name[sortindex[min]].compareTo(name[sortindex[j]])>0)
                min=j;
        }
        if(min!=i){
            int temp=sortindex[min];
            sortindex[min]=sortindex[i];
            sortindex[i]=temp;
        }
    }

    return sortindex;
}
```

2、 `public static int[] SortById(String id[],int N)`

```
public static int[] SortById(String id[],int N){
    int sortindex[]=new int[N];
    for(int i=0;i<N;i++)sortindex[i]=i;

    for(int i=0;i<N;i++){
        int min=i;
        for(int j=i+1;j<N;j++){
            if(id[sortindex[min]].compareTo(id[sortindex[j]])>0)
                min=j;
        }
        if(min!=i){
            int temp=sortindex[min];
            sortindex[min]=sortindex[i];
            sortindex[i]=temp;
        }
    }

    return sortindex;
}
```

3、 `public static int[] SortByHeight(float height[],int N)`

```
public static int[] SortByHeight(float height[],int N){
    int sortindex[]=new int[N];
    for(int i=0;i<N;i++)sortindex[i]=i;

    for(int i=0;i<N;i++){
        int min=i;
        for(int j=i+1;j<N;j++){
            if(height[sortindex[min]]>height[sortindex[j]])
                min=j;
        }
        if(min!=i){
            int temp=sortindex[min];
            sortindex[min]=sortindex[i];
            sortindex[i]=temp;
        }
    }

    return sortindex;
}
```

4、

```
public static int[] SortByWeight(float weight[],int N){
    public static int[] SortByWeight(float weight[],int N){
        int sortindex[]=new int[N];
        for(int i=0;i<N;i++)sortindex[i]=i;

        for(int i=0;i<N;i++){
            int min=i;
            for(int j=i+1;j<N;j++){
                if(weight[sortindex[min]]>weight[sortindex[j]])
                    min=j;
            }
            if(min!=i){
                int temp=sortindex[min];
                sortindex[min]=sortindex[i];
                sortindex[i]=temp;
            }
        }

        return sortindex;
    }
}
```

5、

```
public static int[] SortByBMI(float bmi[],int N){
    public static int[] SortByBMI(float bmi[],int N){
        int sortindex[]=new int[N];
        for(int i=0;i<N;i++)sortindex[i]=i;

        for(int i=0;i<N;i++){
            int min=i;
            for(int j=i+1;j<N;j++){
                if(bmi[sortindex[min]]>bmi[sortindex[j]])
                    min=j;
            }
            if(min!=i){
                int temp=sortindex[min];
                sortindex[min]=sortindex[i];
                sortindex[i]=temp;
            }
        }

        return sortindex;
    }
}
```

5) 在 BMI 类中, 增加 printStudents 函数, 该函数的参数为 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(\n)隔开。

Step1.按要求编写该函数如下:

```
public static void printStudents(String[] ids,String[] names,
    float[] heights,float[] weights,float[] bmis,int N,int sortindex[]){
    System.out.println("排序前:");
    for(int i=0;i<N;i++)
        System.out.printf("%s\t%s\t%.2f\t%.1f\t%.2f\t\n",
            ids[i],names[i],heights[i],weights[i],bmis[i]);

    System.out.println("排序后:");
    for(int i=0;i<N;i++)
        System.out.printf("%s\t%s\t%.2f\t%.1f\t%.2f\t\n",
            ids[sortindex[i]],names[sortindex[i]],heights[sortindex[i]],
            weights[sortindex[i]],bmis[sortindex[i]]);
    System.out.println();
}
```

6) 在 BMI 类的 main 函数中, 调用上述函数, 输入至少 3 名学生, 并按不同属性排序, 并打印排序前、排序后的结果。

Step1: 在 main 中调用如下:

```
public static void main(String[] agrv){
    Scanner input=new Scanner(System.in);

    System.out.println("请输入学生人数N: ");
    int N=input.nextInt();

    String[] ids=new String[N];
    String[] names=new String[N];
    float[] heights=new float[N];
    float[] weights=new float[N];
    float[] bmis=new float[N];
```



```

System.out.println("请输入每个学生的学号、姓名、身高、体重:");
for(int i=0;i<N;i++){
    ids[i]=input.next();
    names[i]=input.next();
    heights[i]=input.nextFloat();
    weights[i]=input.nextFloat();
    bmis[i]=weights[i]/(heights[i]*heights[i]);
}

int sortindex[]=SortByName(names, N);
//int sortindex[]=SortByID(ids, N);
//int sortindex[]=SortByWeight(weights, N);
//int sortindex[]=SortByHeight(heights, N);
//int sortindex[]=SortByBMI(bmis, N);

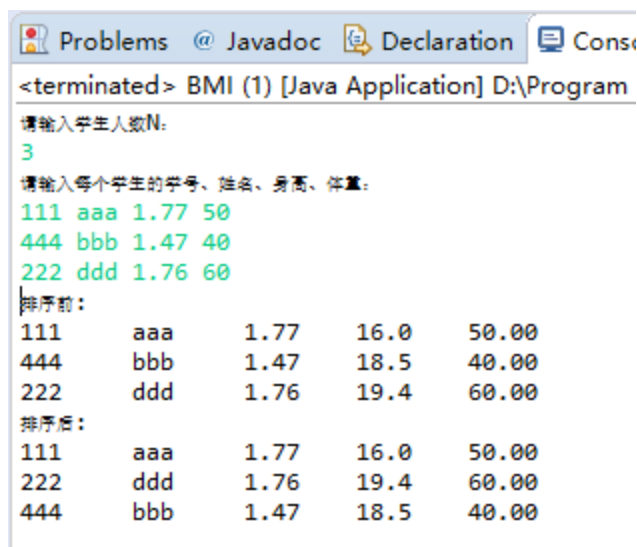
printStudents(ids, names, heights, bmis, weights, N, sortindex);

input.close();
}

```

Step2：排序结果：

1. 按姓名



```

Problems @ Javadoc Declaration Console
<terminated> BMI (1) [Java Application] D:\Program
请输入学生人数N:
3
请输入每个学生的学号、姓名、身高、体重:
111 aaa 1.77 50
444 bbb 1.47 40
222 ddd 1.76 60
排序前:
111    aaa    1.77    16.0    50.00
444    bbb    1.47    18.5    40.00
222    ddd    1.76    19.4    60.00
排序后:
111    aaa    1.77    16.0    50.00
222    ddd    1.76    19.4    60.00
444    bbb    1.47    18.5    40.00

```

2. 按 id

```

Problems @ Javadoc Declaration Cons
<terminated> BMI (1) [Java Application] D:\Program
请输入学生人数N:
3
请输入每个学生的学号、姓名、身高、体重:
111 aaa 1.77 50
222 ddd 1.76 60
444 bbb 1.47 40
排序前:
111    aaa    1.77    16.0    50.00
222    ddd    1.76    19.4    60.00
444    bbb    1.47    18.5    40.00
排序后:
111    aaa    1.77    16.0    50.00
444    bbb    1.47    18.5    40.00
222    ddd    1.76    19.4    60.00

```

3. 按身高

```

Problems @ Javadoc Declaration Co
<terminated> BMI (1) [Java Application] D:\Progra
请输入学生人数N:
3
请输入每个学生的学号、姓名、身高、体重:
111 aaa 1.77 60
222 ddd 1.76 50
333 bbb 1.55 66
排序前:
111    aaa    1.77    19.2    60.00
222    ddd    1.76    16.1    50.00
333    bbb    1.55    27.5    66.00
排序后:
333    bbb    1.55    27.5    66.00
222    ddd    1.76    16.1    50.00
111    aaa    1.77    19.2    60.00

```

4. 按体重

```
Problems @ Javadoc Declaration Conso
<terminated> BMI (1) [Java Application] D:\Program F
请输入学生人数N:
3
请输入每个学生的学号、姓名、身高、体重:
111 aaa 1.77 60
222 ddd 1.76 50
333 bbb 1.55 66
排序前:
111    aaa    1.77    19.2    60.00
222    ddd    1.76    16.1    50.00
333    bbb    1.55    27.5    66.00
排序后:
222    ddd    1.76    16.1    50.00
111    aaa    1.77    19.2    60.00
333    bbb    1.55    27.5    66.00
```

5. 按 bmi 指数

```
Problems @ Javadoc Declaration Cons
<terminated> BMI (1) [Java Application] D:\Program
请输入学生人数N:
3
请输入每个学生的学号、姓名、身高、体重:
111 aaa 1.77 60
222 ddd 1.76 50
333 bbb 1.55 66
排序前:
111    aaa    1.77    19.2    60.00
222    ddd    1.76    16.1    50.00
333    bbb    1.55    27.5    66.00
排序后:
222    ddd    1.76    16.1    50.00
111    aaa    1.77    19.2    60.00
333    bbb    1.55    27.5    66.00
```