实验 1 Java 集合框架

一、实验目的

1. 了解 java 集合框架的层次体系结构；
2. 掌握 List、Set、Map 的基本用法；
3. 掌握集合遍历的基本用法；
4. 了解 Stream 的基本用法。

二、实验内容

以下各个程序，请输入电脑，观察程序输出

1、List 的应用

```
package cn.mldn.demo;
import java.util.ArrayList;
import java.util.List;
public class JavaCollectDemo {
    public static void main(String[] args) {
        List<String> all = new ArrayList<String>();    // 为List父接口进行实例化
        all.add("Java");                               // 保存数据
        all.add("Java");                               // 保存重复数据
        all.add("www.scau.edu.cn");                    // 保存数据
        all.add("张老师");                              // 保存数据
        all.forEach((str) -> {                         // 集合输出
        System.out.print(str + "、");
        });
    }
}
```

2、Set 的应用

```
package cn.mldn.demo;
import java.util.Set;
import java.util.TreeSet;
public class JavaCollectDemo {
    public static void main(String[] args) {
        Set<String> all = new TreeSet<String>();       // 为Set父接口进行实例化
        all.add("java");                               // 保存数据
        all.add("Java");                               // 保存重复数据
        all.add("Java");                               // 保存重复数据
        all.add("Hello");                              // 保存数据
        System.out.println(all);                       // 直接输出集合对象
```

```
    }
}
```

3、Map 的应用

```
public class JavaCollectDemo {
    public static void main(String[] args) {
        Map<String, Integer> map = new HashMap<String, Integer>(); // 创建Map集合
        map.put("one", 1);                                          // 保存数据
        map.put("two", 2);                                          // 保存数据
        map.put("one", 101);                                  // key重复，发生覆盖
        map.put(null, 0);                                           // key为null
        map.put("zero", null);                                      // value为null
        System.out.println(map.get("one"));                         // key存在
        System.out.println(map.get(null));                          // key存在
        System.out.println(map.get("ten"));                         // key不存在
    }
}
```

4、Stack 的应用

```
ublic class JavaCollectDemo {
    public static void main(String[] args) {
        Stack<String> all = new Stack<String>() ;          // 实例化栈结构
        all.push("A") ;                                    // 入栈操作
        all.push("B") ;                                    // 入栈操作
        all.push("C") ;                                    // 入栈操作
        System.out.println(all.pop());                     // 出栈操作
        System.out.println(all.pop());                     // 出栈操作
        System.out.println(all.pop());                     // 出栈操作
        System.out.println(all.pop());            // 无数据、EmptyStackException
    }
}
```

5、Properties 的应用

```
public class JavaCollectDemo {
    public static void main(String[] args) {
        Properties prop = new Properties();                    // 属性存储
        prop.setProperty("scau", " www.scau.edu.cn ");         // 设置属性内容
        prop.setProperty("scaujava", "www.scau.edu.cn ");      // 设置属性内容
        System.out.println(prop.getProperty("scau"));          // 根据key查找属性
        System.out.println(prop.getProperty("yootk", "NoFound"));// 根据key查找属性
        System.out.println(prop.getProperty("yootk"));         // 根据key查找属性
```

```
    }
}
```

6、迭代遍历

```
public class JavaCollectDemo {
    public static void main(String[] args) {
        Set<String> all = Set.of("Hello", "ScauJava", "Scau");      // 创建Set集合
        Iterator<String> iter = all.iterator();        // 实例化Iterator接口对象
        while (iter.hasNext()) {                                    // 集合是否有数据
            String str = iter.next();                               // 获取每一个数据
            System.out.print(str + "、");                           // 输出数据
        }
    }
}
```

7、双向迭代输出

```
public class JavaCollectDemo {
    public static void main(String[] args) {
        List<String> all = new ArrayList<String>();         // 为List父接口进行实例化
        all.add("小李老师");                                  // 保存数据
        all.add("Java");                                    // 保存数据
        all.add("www.scau.edu.cn");                         // 保存数据
        ListIterator<String> iter = all.listIterator() ;// 获取ListIterator接口实例
        System.out.print("由前向后输出：");
        while(iter.hasNext()) {                             // 由前向后迭代
            System.out.print(iter.next() + "、");
        }
        System.out.print("\n由后向前输出：");
        while (iter.hasPrevious()) {                        // 由后向前迭代
            System.out.print(iter.previous() + "、");
        }
    }
}
```

8、枚举输出

```
public class JavaCollectDemo {
    public static void main(String[] args) {
        Vector<String> all = new Vector<String>();          // 实例化Vector
        all.add("小李老师");                                  // 保存数据
        all.add("ScauJava");                                // 保存数据
        all.add("www. scau.edu.cn ");                       // 保存数据
```

```
        Enumeration<String> enu = all.elements() ;          // 获取Enumeration实例
        while (enu.hasMoreElements()) {
            String str = enu.nextElement() ;
            System.out.print(str + "、");
        }
    }
}
```

9、foreach 输出

```
public class JavaCollectDemo {
    public static void main(String[] args) {
        Set<String> all = new HashSet<String>();        // 实例化Set
        all.add("老师");                                 // 保存数据
        all.add("Java");                                 // 保存数据
        all.add("www.scau.edu.cn ");                     // 保存数据
        for (String str : all) {
            System.out.print(str + "、");
        }
    }
}
```

10、使用 Stream 进行数据采集

```
public class JavaCollectDemo {
    public static void main(String[] args) throws Exception {
        List<String> all = new ArrayList<String>();             // 实例化List集合
        Collections.addAll(all, "Java", "JavaScript", "JSP",
                "Json", "Python", "Ruby", "Go");                // 集合数据保存
        Stream<String> stream = all.stream();                   // 获取Stream接口对象
        // 将每一个元素全部变为小写字母，而后查询是否存在有字母“j”，如果存在则进行个数统计
        System.out.println(stream.filter((ele) ->
    ele.toLowerCase().contains("j")).count());
    }
}
```

11、使用 Stream 进行数据采集

```
public class JavaCollectDemo {
    public static void main(String[] args) throws Exception {
        List<String> all = new ArrayList<String>();                 // 实例化List集合
        Collections.addAll(all, "Java", "JavaScript", "JSP",
                "Json", "Python", "Ruby", "Go");                    // 集合数据保存
```

```
        Stream<String> stream = all.stream();          // 获取Stream接口对象
        // 获取元素中包含有字母“j”的数据，利用skip()跳过2个数据，利用limit()取出2个数据
        List<String> result = stream.filter((ele) ->
ele.toLowerCase().contains("j"))
            .skip(2).limit(2).collect(Collectors.toList());    // 获取处理后的数据
        System.out.println(result);
    }
}
```

三、根据下面要求编写程序（二选一）

1、实现简单学生信息管理系统，要求如下：

（1）用 Collection 中 List 实现一个简单的学生信息管理系统。学生信息有：学号、姓名、年龄、三门课成绩等。在其上实现增删改查的操作；

（2）用 Map 实现电话簿 管理程序。根据姓名查询电话号码。

2、编写一个实现 IOC 功能的简单 Spring 框架，包含对象注册、对象管理、及暴露给外部获取对象的功能，并编写测试程序。扩展注册器的方式，要求采用 XML 和 txt 文件。具体参考课本 2.6。

四、实验结果

写实验报告。内容包括：

1. 习题的运行结果，源程序。
2. 程序调试中出现的错误提示。（英文、中文对照）
3. 若有没通过的程序，分析原因。