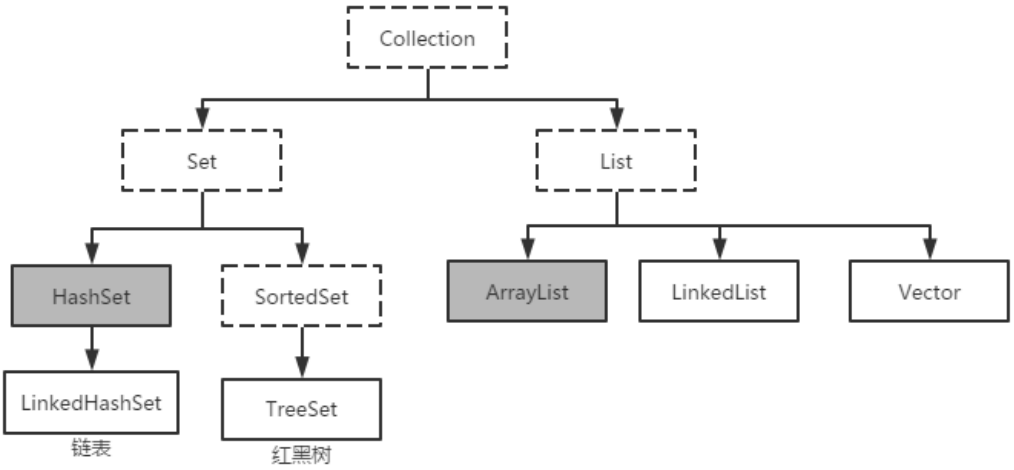
集合



Set

HashSet

Set存储一组对象，不保证按添加的顺序来存放（根据hashcode决定顺序），相同（hashcode相同，equals返回true）的对象会被覆盖

默认HashSet的初始长度是16，负载因子是0.75。初始长度可以用int参数的构造方法来设置

遍历Set的方式：

- foreach
- 迭代器（iterator）
- 朗姆达表达式（Lambda）,也称为箭头函数

泛型：用于指定集合里面存放对象的类型，写在类型后面的尖括号里面

```
package com.hqyj;

import java.util.HashSet;
import java.util.Iterator;
import java.util.Set;

public class HashSetDemo {

    public static void main(String[] args) {
        //定义set对象
        //默认长度是16，当存储容量超过0.75（负载因子）时自动增长（按默认长度增长16）
        //存入的顺序跟取出来的顺序可能不一样
        //相同（hashcode相同，equals方法返回true）的对象保存会覆盖掉原来的
        Set<String> set = new HashSet<>();
        //添加数据到set
        set.add("Java基础");
        set.add("Java面向对象");
```

```
        set.add("Java高级");
        set.add("Java高级");
        System.out.println("Java基础".hashCode() + ", " + "Java面向对象".hashCode()
+ ", " +"Java高级".hashCode());
        //foreach 遍历读取set中的数据
        for(String obj : set) {
            System.out.println(obj);
        }

        System.out.println("============================");

        //用迭代器遍历Set集合
        Iterator<String> iterator = set.iterator();
        while(iterator.hasNext()) {
            System.out.println(iterator.next());
        }

        System.out.println("============================");

        //朗姆达（Lambda）表达式（箭头函数）遍历
        set.forEach((str)->{
            System.out.println(str);
        });
        //朗姆达简化写法，用::。   双冒号后面是方法名称
        set.forEach(System.out::println);

    }
}
```

HashSet存放自定义对象，重写hashcode和equals方法

实体类Student

```
package com.hqyj;

public class Student {
    private String stuNum; //学号
    private String stuName;  //姓名

    public Student() {
        super();
    }
    public Student(String stuNum, String stuName) {
        super();
        this.stuNum = stuNum;
        this.stuName = stuName;
    }
    public String getStuNum() {
        return stuNum;
    }
    public void setStuNum(String stuNum) {
        this.stuNum = stuNum;
    }
```

```
    public String getStuName() {
        return stuName;
    }
    public void setStuName(String stuName) {
        this.stuName = stuName;
    }
    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + ((stuNum == null) ? 0 : stuNum.hashCode());
        return result;
    }
    @Override
    public boolean equals(Object obj) {

        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Student other = (Student) obj;
        if (stuNum == null) {
            if (other.stuNum != null)
                return false;
        } else if (!stuNum.equals(other.stuNum))
            return false;
        return true;

//      return false;
    }
    @Override
    public String toString() {
        return "Student [stuNum=" + stuNum + ", stuName=" + stuName + "]";
    }
}
```

使用自定义类

```
package com.hqyj;

import java.util.HashSet;
import java.util.Set;

public class SetStudentDemo {

    public static void main(String[] args) {
        Set<Student> set = new HashSet<>();
        set.add(new Student("123","张三"));
        set.add(new Student("124","李四"));
        set.add(new Student("125","王五"));
        set.add(new Student("126","狗蛋"));
        set.add(new Student("126","狗蛋"));

        for(Student stu : set) {
```

```
            System.out.println(stu);
        }
    }

}
```

LinkedHashSet

是HashSet的子类，跟HashSet类似，它是用链表方式实现，按添加的顺序保存

```
package com.hqyj;

import java.util.LinkedHashSet;
import java.util.Set;

public class LinkedHashSetDemo {

    public static void main(String[] args) {
        //用链表方式实现，按存入的顺序保存。其它的跟HashSet完全一样
        Set<String> set = new LinkedHashSet<>();
        set.add("Java基础");
        set.add("Java面向对象");
        set.add("Java高级");

        for(String s : set) {
            System.out.println(s);
        }

    }

}
```

TreeSet

用红黑树方式实现，保存的元素会自动排序

要求存入的类型必须实现Comparable接口，否则运行时会报类型转换错误

比Set多提供一些方法，比如：first(), last(), lower(), higher()

```
package com.hqyj;

import java.util.TreeSet;

public class TreeSetDemo {

    public static void main(String[] args) {
        //TreeSet是用红黑树方式实现，它会添加进集合数据进行排序
        //要求存放的对象实现Comparable接口,否则运行时要报ClassCastException异常
        TreeSet<Integer> set = new TreeSet<>();
        set.add(15);
        set.add(2);
        set.add(8);
        set.add(6);
```

```
        for(int n : set) {
            System.out.println(n);
        }
        /*
        TreeSet<Student> set2 = new TreeSet<>();
        set2.add(new Student("1111","Tom"));
        for(Student stu : set2) {
            System.out.println(stu);
        }*/
        System.out.println("====================");
        System.out.println(set.first());  //第一个元素
        System.out.println(set.last());  //最后一个元素
        System.out.println(set.lower(9));  //返回比指定值小的最大的元素
        System.out.println(set.higher(5));  //返回比指定值大的最小元素
    }

}
```

List

按添加的顺序保存

可以存储重复元素

可以用下标来随机访问

ArrayList

用数组方式实现，默认长度是10

```
package com.hqyj;

import java.util.ArrayList;
import java.util.List;

public class ArrayListDemo {

    public static void main(String[] args) {
        //按添加顺序保存
        //可以保存重复的元素
        //可以用下标随机访问
        //用数组方式实现，默认长度是10
        //线程不安全
        List<String> list = new ArrayList<>();
        list.add("Java基础");
        list.add("Java面向对象");
        list.add("Java高级");
        list.add("Java高级");

        for(String s : list) {
            System.out.println(s);
        }

        System.out.println("==================");
```

```
        System.out.println(list.get(0));  //按下标来读取元素，下标类似数组
        System.out.println(list.size());  //列表长度

        System.out.println(list.indexOf("Java高级")); //返回指定元素第一次出现的下标
        System.out.println(list.lastIndexOf("Java高级")); //返回指定元素最后一次出现的
下标
        List<String> subList = list.subList(0, 2); //返回从第一个参数开始，到第二参数
结束的子list，左闭右开的区间
        subList.forEach(System.out::println);

    }

}
```

LinkedList

用链表（插入和删除速度快，访问速度满）实现

Vector

功能同ArrayList, 它是一个老旧的类，不推荐使用。它是线程安全的

今日作业

1. ArrayList的四种遍历方式（for， foreach，iterator，Lambda）
2. 定义一个int数组，初始化一些数据（有重复）。去重，排序（升序），打印出来
3. 完成学生管理功能（增删改查 CRUD）