java.utils.Collections集合工具类

**//public static** <T> **boolean** addAll(Collection<T> c,T elments)向集合中添加一些元素

**import** java.util.ArrayList;

**import** java.util.Collections;

**public class** VarArgs {

**public static void** main(String[] args) {

ArrayList<String> list=**new** ArrayList<>();

*/\* list.add("a");*

*list.add("b");*

*list.add("c");*

*list.add("d");*

*list.add("e");\*/*

Collections.*addAll*(list,**"a"**,**"b"**,**"c"**);

}

}

*/\*public static void shuffle(List<?> List)：打乱集合顺序*

*\* \*/*

**public class** VarArgs {

**public static void** main(String[] args) {

ArrayList<String> list=**new** ArrayList<>();

Collections.*addAll*(list,**"a"**,**"b"**,**"c"**,**"d"**);

Collections.*shuffle*(list);

}

**import** java.util.ArrayList;

**import** java.util.Collections;

*/\*public static <T> void sort(List<?> list）*

*集合元素中按照默认规则排序*

*\* \*/*

**public class** VarArgs {

**public static void** main(String[] args) {

ArrayList<Integer> list1=**new** ArrayList<>();

Collections.*addAll*(list1,3,4,6,1);

Collections.*sort*(list1); *//升序，也可对字符串排序*

System.***out***.println(list1);

ArrayList<Person> people=**new** ArrayList<>();

people.add(**new** Person(**"Mark"**,18));

people.add(**new** Person(**"Jack"**,20));

Collections.*sort*(people);

System.***out***.println(people);

*/\**

*\* 对对象排序，必须实现comparable接口，重写compareTo方法定义排序规则*

*\* this-参数 升序*

*\* \*/*

}

**import** java.util.ArrayList;

**import** java.util.Collections;

**import** java.util.Comparator;

*/\*pubic static <T> void sort(List<T> list,Comparator<? super T>*

*将集合按照指定规则排序*

*Compartor和Compareble区别*

*Comparable自己与参数比较，自己要实现Comparable接口，重写比较规则CompareTo方法*

*Comparator相当于找了个第三方裁判，比较两个*

*\* \*/*

**public class** VarArgs {

**public static void** main(String[] args) {

ArrayList<Integer> list1=**new** ArrayList<>();

Collections.*addAll*(list1,3,4,6,1);

Collections.*sort*(list1, **new** Comparator<Integer>() {

@Override

*//重写比较的规则*

**public int** compare(Integer o1, Integer o2) {

**return** o1-o2; *//升序*

}

});

ArrayList<Person> people=**new** ArrayList<>();

Collections.*addAll*(people,**new** Person(**"Max"**,18),**new** Person(**"Kate"**,30));

Collections.*sort*(people, **new** Comparator<Person>() {

@Override

//如果年龄相同 按照name首字母排序

**public int** compare(Person o1, Person o2) {

**int** result= o1.getAge()-o2.getAge();

**if** (result==0){

result=o1.getName().charAt(0)-o2.getName().charAt(0);

}

*//升序*

**return** result;

}

});

System.***out***.println(list1);

System.***out***.println(people);

}

}

**public class** Person **implements** Comparable<Person> { 实现comparable接口

**public int** compareTo(Person person){ 重写compareTo方法

**return this**.getAge()-person.getAge();*//升序*

*//public int compareTo(Person person){*

*return Double.compare(age,person.age)}*

*/\*return person.getAge()-this.getAge()降序*

*\* \*/*

}