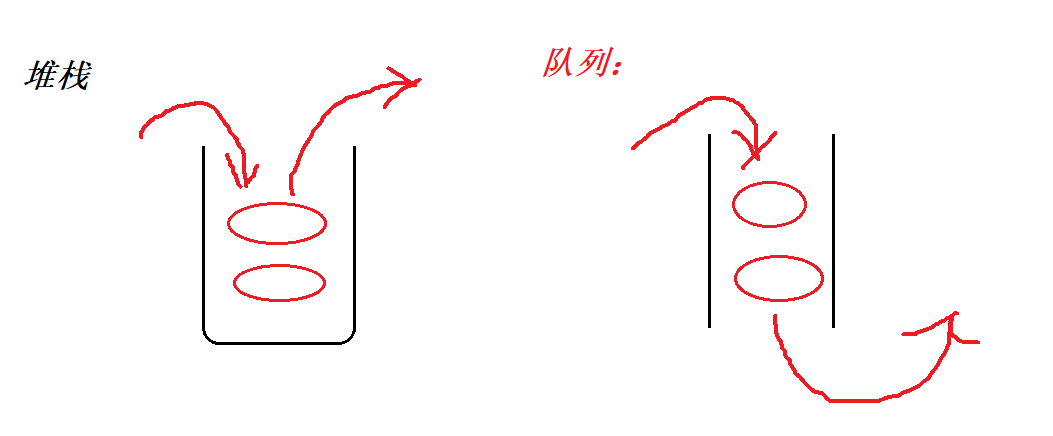
**`**

**容器作业**

1. **填空题**
2. **Java集合框架提供了一套性能优良、使用方便的接口和类，包括Collection和Map两大类，它们都位于 java.util 包中**
3. **队列和堆栈有些相似，不同之处在于 堆栈：先进后出，后进先出；队列 ：先进先出 。**



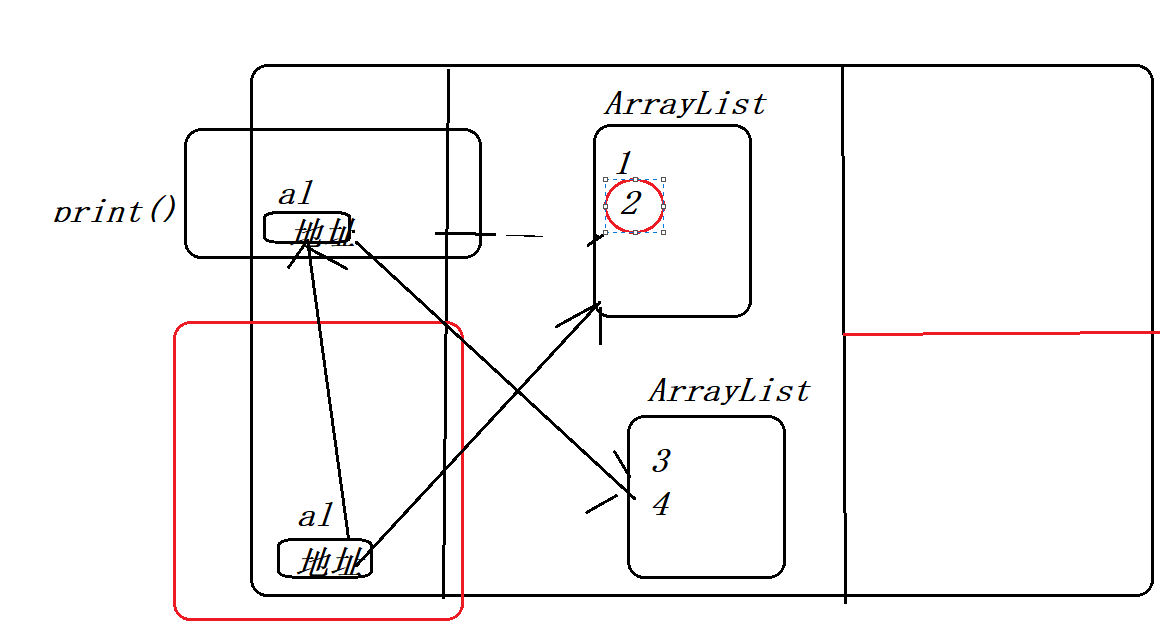
1. **链表 结构是一种由多个节点组成的线性数据结构，并且每个节点包含有数据以及指向下一个节点的引用。**
2. **\_\_\_LinkedList\_\_\_\_\_\_\_\_\_\_\_是一种集合类，它 采用链表作为的存储结构，便于删除和添加元素，但是按照索引查询元素效率低下。**
3. **TreeSet 是一种Collection类型的集合类，其中元素唯一，并采用二叉树作为存储结构，元素按照自然顺序排列。**
4. **如果希望将自定义类Student的多个对象放入集合TreeSet，实现所有元素按照某个属性的自然顺序排列，则需要Student类实现\_\_\_\_\_Comparable\_\_\_\_\_\_\_\_\_\_\_\_\_接口。**
5. **在Java中 HashMap 集合的访问时间接近稳定，它是一种键值对映射的数据结构。这个数据结构是通过数组来实现的。**
6. **迭代器Iterator为集合而生，专门实现集合遍历，该接口有三个方法，分别是hasNext() 、\_\_\_\_next()\_\_\_\_\_\_\_\_、remove()。**
7. **选择题**

|  |  |  |
| --- | --- | --- |
| **1.** | **以下选项中关于Java集合的说法错误的是（ AC ）。（选择二项）** | |
|  |  |  |
|  | **A.** | **List接口和Set接口是Collections接口有两个子接口** |
|  | **B.** | **List接口中存放的元素具有有序，不唯一的特点** |
|  | **C.** | **Set接口中存放的元素具有无序，不唯一的特点** |
|  | **D.** | **Map接口存放的是映射信息，每个元素都是一个键值对** |

|  |  |  |
| --- | --- | --- |
| **2.** | **如下Java代码，输出的运行结果是（ A ）。（选择一项）** | |
|  | **public class Test {**  **public static void main(String[ ] args) {**  **List<String> list=new ArrayList<String>();**  **list.add("str1");**  **list.add(2, "str2"); //该add方法在添加元素时，要求该数组中当前的位置已经存储了元素，才能添加**  **String s=list.get(1);**  **System.*out*.println(s);**  **}**  **}** | |
|  |  |  |
|  | **A** | **运行时出现异常** |
|  | **B.** | **正确运行，输出str1** |
|  | **C.** | **正确运行，输出str2** |
|  | **D.** | **编译时出现异常** |

|  |  |  |
| --- | --- | --- |
| **3.** | **以下Java代码的作用是首先将一个数组的内容存入集合，然后判断集合中是否有指定的元素存在，其中共有（ D ）处错误。（选择一项）** | |
|  | **import java.util.List;**  **public class Test {**  **public int getIndexofArray(float[] f){**  **int rtn=-1;**  **float objf=3.4;**  **List list=null;**  **for(int i=0;i<f.size( );i++){**  **list.add(f[i]);**  **}**  **for(int i=0;i<list.size( );i++){**  **float tmp=(float)list.get(i);**  **if(objf==tmp){**  **rtn=i;**  **}**  **}**  **return rtn;**  **}**  **}** | |
|  |  |  |
|  | **A** | **0** |
|  | **B.** | **1** |
|  | **C.** | **2** |
|  | **D.** | **3** |

|  |  |  |
| --- | --- | --- |
| **4.** | **分析如下Java 代码，编译运行后将输出（ B ）。（选择一项）** | |
|  | **public class Test {**  **public Test() {**  **}**  **static void print(List<Integer> al) {**  **al.add(2);**  **al = new ArrayList<Integer>();**  **al.add(3);**  **al.add(4);**  **}**  **public static void main(String[] args) {**  **List<Integer> al = new ArrayList<Integer>();**  **al.add(1);**  ***print*(al);**  **System.*out*.println(al.get(1));**  **}**  **}** | |
|  |  |  |
|  | **A** | **1** |
|  | **B.** | **2** |
|  | **C.** | **3** |
|  | **D.** | **4** |



|  |  |  |
| --- | --- | --- |
| **5.** | **在Java中,下列集合类型可以存储无序、不重复的数据的是（ D ）。（选择一项）** | |
|  |  |  |
|  | **A** | **ArrayList** |
|  | **B.** | **LinkedList** |
|  | **C.** | **TreeSet** |
|  | **D.** | **HashSet** |

|  |  |  |
| --- | --- | --- |
| **6.** | **以下代码的执行结果是（ C ）。（选择一项）** | |
|  | **Set<String> s=new HashSet<String>();**  **s.add("abc");**  **s.add("abc");**  **s.add("abcd");**  **s.add("ABC");**  **System.*out*.println(s.size());** | |
|  |  |  |
|  | **A.** | **1** |
|  | **B.** | **2** |
|  | **C.** | **3** |
|  | **D.** | **4** |

|  |  |  |
| --- | --- | --- |
| **7.** | **给定如下Java代码，编译运行的结果是（ C ）。（选择一项）** | |
|  | **public class Test {**  **public static void main(String[] args) {**  **Map<String, String> map = new HashMap<String, String>();**  **String s = "code";**  **map.put(s, "1");**  **map.put(s, "2");**  **System.*out*.println(map.size());**  **}**  **}** | |
|  |  |  |
|  | **A** | **编译时发生错误** |
|  | **B.** | **运行时引发异常** |
|  | **C.** | **正确运行，输出：1** |
|  | **D.** | **正确运行，输出：2** |

|  |  |  |  |
| --- | --- | --- | --- |
| **8.** | **下面集合类中属于非线程安全，且结构采用了哈希表的是（ C ）。（选择一项）** | | |
|  |  | |  |
|  | **A.** | | **Vector** |
|  | **B.** | | **ArrayList** |
|  | **C.** | | **HashMap** |
|  | **D.** | | **Hashtable** |
| **9.** | **在Java中，LinkedList类与ArrayList类同属于集合框架类，下列（ C ）选项中是LinkedList类有而ArrayList类没有的方法。（选择两项）** | | |
|  |  |  | |
|  | **A** | **add(Object o)** | |
|  | **B.** | **add(int index,Object o)** | |
|  | **C.** | **getFirst()** | |
|  | **D.** | **removeLast()** | |

1. **判断题**
2. **数组和集合中的元素可以是任何数据类型，包括基本类型和引用类型。( X )**

**集合在存储基本数据类型时，首先进行包装，变成引用类型，然后再添加；**

1. **容器（集合）指的是“可以容纳其他对象的对象”。（ √ ）**
2. **Java集合中的Set接口和List接口都是从Collection接口派生出来的。（ √ ）**
3. **Collection 接口存储一组不唯一，有序的对象，它有两个子接口：List和Set。（ × ）**
4. **Collection是Java集合顶级接口，其中的元素无序，唯一。Java平台不提供这个接口任何直接的实现。（ × ）**
5. **List是有序的Collection，使用此接口能够精确的控制每个元素插入的位置。用户能够使用索引来访问List中的无素，这类似于Java的数组。（ × ）**

**ArrayList**

**LinkedList**

1. **HashSet采用哈希表存储结构，特点是查询速度快，但是其中元素无序排列。（ √ ）**
2. **LinkedHashMap是一种有序的HashMap，查询速度快，便于添加删除操作。（ √ ）**
3. **基本数据类型的值可以被直接存储在Vector对象中。（ × ） //需要包装**
4. **Dictionary建立了关键字和值的映射，只要提供一个关键字，Dictionary就会返回一个相应的值。（ √ ）**

**版本**

**Dictionary JDK1.0**

**Map JDK1.2**

1. **泛型是JavaSE1.7的新特性，泛型的本质是参数化类型，也就是说所操作的数据类型被指定为一个参数。Java语言引入泛型的好处是安全简单。（ × ）**

**泛型是JavaSE1.5的新特性;**

1. **Collection是专门操作集合的工具类，提供一系列静态方法实现对各种集合操作。（ × ）**
2. **Iterator接口可以遍历任何Collection接口的实现类，可以从一个Collection中使用iterator( )方法来获取迭代器实例。迭代器取代了Java集合框架中的Enumeration。（ √ ）**
3. **采用增强for循环遍历List或者Set，如果List或者Set没有加泛型，也能遍历。（ √ ）**
4. **在类已经重写equals和hashCode方法的前提下，equals返回true，hashcode一定相等。（ √ ）**
5. **简答题**
6. **集合和数组的比较 ：**

**集合：动态扩容 泛型之前，添加数据类型：引用，基本（包装）**

**泛型之后，添加泛型要求的数据类型；**

**数组：不能改变 ，只能添加指定的数据类型；**

1. **简述List、Set、Collection、Map的区别和联系。**
2. **ArrayList和LinkedList的区别和联系。它们的底层分别是用什么实现的？**
3. **HashSet采用了哈希表作为存储结构，请说明哈希表的特点和实现原理。**

**提示：结合Object类的hashCode()和equals()说明其原理**

1. **Vector和ArrayList的区别和联系。**
2. **请你简述HashMap和Hashtable的区别？**
3. **说明isEmpty的作用，并说明下面代码有问题吗?**

|  |
| --- |
| **Collection c = null;**  **System.out.println(c.isEmpty());** |

1. **写出List、Set、Map中使用泛型的例子。**
2. **使用泛型有什么好处?**
3. **每个对象都有一个哈希码吗？哈希码是根据什么生成的？会不会重复?**
4. **编码题**
5. **使用List和Map存放多个图书信息，遍历并输出。其中商品属性：编号，名称，单价，出版社；使用商品编号作为Map中的key。**

|  |
| --- |
| **class Book{  private int id;  private String name;  private double price;  private String press;   public Book() {  }   public Book(int id, String name, double price, String press) {  this.id = id;  this.name = name;  this.price = price;  this.press = press;  }   @Override  public String toString() {  return "Book{" +  "id=" + id +  ", name='" + name + '\'' +  ", price=" + price +  ", press='" + press + '\'' +  '}';  }   public int getId() {  return id;  }   public String getName() {  return name;  }   public double getPrice() {  return price;  }   public String getPress() {  return press;  } } public class homeWork {  public static void main(String[] args) {   //0.构建对象：  Book book1 = new Book(3, "流量地球3", 50.5, "人民出版社");  Book book2 = new Book(1, "流量地球1", 60.5, "人民出版社");  Book book3 = new Book(2, "流量地球2", 40.5, "人民出版社");  //1.list存储：  List<Book> list = new ArrayList<Book>();  list.add(book1);  list.add(book2);  list.add(book3);   //2.map存储：自定义排序  TreeMap<Integer, Book> map = new TreeMap<Integer, Book>(new Comparator<Integer>() {  @Override  public int compare(Integer o1, Integer o2) {  return o2 - o1; //id降序  }  });   map.put(book1.getId() , book1);  map.put(book2.getId() , book2);  map.put(book3.getId() , book3);   //遍历：  Set<Map.Entry<Integer, Book>> set = map.entrySet();  Iterator<Map.Entry<Integer, Book>> iter = set.iterator();  while(iter.hasNext()){  Map.Entry<Integer, Book> entry = iter.next();  System.out.println(entry.getKey()+","+entry.getValue());  }  } }** |

1. **使用HashSet和TreeSet存储多个商品信息，遍历并输出；其中商品属性：编号，名称，单价，出版社；要求向其中添加多个相同的商品，验证集合中元素的唯一性。**

**提示：向HashSet中添加自定义类的对象信息，需要重写hashCode和equals( )**

**向TreeSet中添加自定义类的对象信息，需要实现Comparable接口，指定比较规则**

1. **实现List和Map数据的转换。具体要求如下：**

**功能1：定义方法public void listToMap( ){ }将List中Student元素封装到Map中**

* + 1. **使用构造方法Student(int id,String name,int age,String sex )创建多个学生信息并加入List**
    2. **遍历List，输出每个Student信息**
    3. **将List中数据放入Map，使用Student的id属性作为key，使用Student对象信息作为value**
    4. **遍历Map，输出每个Entry的key和value**

**功能2：定义方法public void mapToList( ){ }将Map中Student映射信息封装到List**

1. **创建实体类StudentEntry，可以存储Map中每个Entry的信息**
2. **使用构造方法Student(int id,String name,int age,String sex )创建多个学生信息，并使用Student的id属性作为key，存入Map**
3. **创建List对象，每个元素类型是StudentEntry**
4. **将Map中每个Entry信息放入List对象**

**功能3：说明Comparable接口的作用，并通过分数来对学生进行排序。**

1. **用代码写出遍历List的三种方式。**
2. **用代码写出遍历Set的两种方式。**
3. **用代码写出遍历map的方式。**
4. **可选题**
5. **[假如有以下email数据“aa@sohu.com,bb@163.com,cc@sina.com,..”现需要把email中的用户部分和邮件地址部分分离，分离后以键值对应的方式放入HashMap？](mailto:假如有以下email数据\“aa@sohu.com,bb@163.com,cc@sina.com,..\”现需要把email中的用户部分和邮件地址部分分离，分离后以键值对应的方式放入HashMap？)**

|  |
| --- |
| **String str = "aa@sohu.com,bb@163.com,cc@sina.com"; //切割： String[] split = str.split(",");  HashMap<String, String> map = new HashMap<>(); //通过编译拿到每一个邮箱： for(String email :split){  //aa sohu.com  String[] split1 = email.split("@");  //存储：  map.put(split1[0],split1[1]); }** |

1. **由控制台按照固定格式输入学生信息，包括学号，姓名，年龄信息，当输入的内容为exit退出；将输入的学生信息分别封装到一个Student对象中，再将每个Student对象加入到一个集合中，要求集合中的元素按照年龄大小正序排序；最后遍历集合，将集合中学生信息写入到记事本，每个学生数据占单独一行。**

**推荐步骤：**

* 1. **创建Student类，并指定按照年龄正序排列**
  2. **通过控制台输入多个不同Student信息。格式规定为：编号#姓名#年龄**
  3. **取出字符串中相应信息放入Student对象，并将Student加入到集合中**
  4. **遍历集合的过程中将学生的信息输入到记事本**

**难点：**

* 1. **如何指定学生按照年龄正序排列**
  2. **如果从字符串“编号#姓名#年龄”中提取学生信息**
  3. **放入哪种集合后可以保证学生按照年龄大小正序排列**
  4. **如何将集合中学生信息写入记事本，每个学生数据占单独一行**

|  |
| --- |
| **class Student implements Comparable<Student>{  private int id;  private String name;  private int age;   public Student(int id, String name, int age) {  this.id = id;  this.name = name;  this.age = age;  }   /\*\*  \* 年龄升序：  \*/  @Override  public int compareTo(Student o) {  return this.age - o.age;  }   @Override  public String toString() {  return "Student{" +  "id=" + id +  ", name='" + name + '\'' +  ", age=" + age +  '}';  } }  public class HomeWork{  public static void main(String[] args) {   Scanner input = new Scanner(System.in);  TreeSet<Student> set = new TreeSet<Student>();   while (true){  System.out.println("输入格式：");  String info = input.nextLine();  if ("exit".equals(info)){  break;  }  //切割：  String[] fields = info.split("#");  //封装对象：  Student s = new Student(Integer.parseInt(fields[0]),  fields[1],  Integer.parseInt(fields[2]));  set.add(s);  }   Iterator<Student> iter = set.iterator();  while(iter.hasNext()){  Student student = iter.next();  System.out.println(student);  }   } }** |

1. **针对List中新增的有关顺序的方法，如add(int index, E element) , get(int index)等每个都进行测试。并且使用debug来帮助我们理解程序运行。**
2. **Collection和Collections有什么区别?**
3. **Map中，key能否重复?如果重复，会有什么现象?**
4. **请你简述Set和List的特点跟区别?**
5. **我有一些数据，需要频繁的查询，插入和删除操作非常少，并且没有线程之间的共享，使用List下面的哪个实现类好一些?**
6. **我想定义一个数组。该数组既可以放：Dog对象、也可以放Cat对象、还可以放Integer对象，怎么定义?**