**题目1：List接口有哪些常见的实现类，它们的区别是什么，请详细描述。**

**答案：**

|  |
| --- |
| 1. ArrayList是最常用的List实现类，内部是通过数组实现的，它允许对元素进行快速随机访问。 数组的缺点是每个元素之间不能有间隔，当数组大小不满足时需要增加存储能力，就要将已经有数组的数据复制到新的存储空间中。 ...  2. Vector与ArrayList一样，也是通过数组实现的，不同的是它支持线程的同步，即某一时刻只有一个线程能够写Vector，避免多线程同时写而引起的不一致性，但实现同步需要很高的花费，因此，访问它比访问ArrayList慢。  3. LinkedList是用链表结构存储数据的，很适合数据的动态插入和删除，随机访问和遍历速度比较慢。 另外，接口中没有定义的方法get，remove，insertList，专门用于操作表头和表尾元素，可以当作堆栈、队列和双向队列使用 |

**题目2：现有接口ISleep如下，请使用匿名内部类的方式创建一个对象，然后借助该对象调用sleep()方法。**



**答案：**

|  |
| --- |
| ISleep sleeper = new ISleep() {  @Override  public void sleep() {  try {  Thread.sleep(1000);  } catch (InterruptedException e) {  e.printStackTrace();  }  }  };  sleeper.sleep();  ... |

**题目3：请定义一个Student类，包含学号id和分数score属性，创建多个对象插入TreeSet集合对象，要求插入时按score降序，如果score相同则按id升序。**

**答案：**

|  |
| --- |
| import java.util.Comparator;  import java.util.TreeSet;  class Student {  int id;  int score;  public Student(int id, int score) {  this.id = id;  this.score = score;  }  @Override  public String toString() {  return "Student{" +  "id=" + id +  ", score=" + score +  '}';  }  @Override  public int compareTo(Object o) {  if (this == o) return 0;  if (o == null || getClass() != o.getClass()) return -1;  Student student = (Student) o;  if (score != student.score) {  return student.score - score;  } else {  return id - student.id;  }  }  }  public class Main {  public static void main(String[] args) {  TreeSet<Student> students = new TreeSet<>(new Comparator<Student>() {  @Override  public int compare(Student o1, Student o2) {  return o2.compareTo(o1);  }  });  students.add(new Student(1, 90));  students.add(new Student(2, 80));  students.add(new Student(3, 80));  students.add(new Student(4, 70));  for (Student student : students) {  System.out.println(student);  }  }  }  ... |