1、填空

Collection 接口的特点是元素是 对象 ；

List 接口的特点是元素 有（有|无）顺序， 可以 （可以|不可以）重复；

Set 接口的特点是元素 无 （有|无）顺序， 不可以 （可以|不可以）重复；

1. （List）有如下代码

**import** java.util.\*;

**public** **class** Demo {

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

List list = **new** LinkedList();

list.add("Hello");

list.add("World");

list.add(1, "Learn");

list.add(1, "Java");

*printList*(list);

}

**public** **static** **void** printList(List list) {

//System.out.println(list);,

**for** (**int** i = 0; i < list.size(); i++) {

System.***out***.println(list.get(i));

}

**for** (Object o : list) {

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

}

Iterator itor = list.iterator();

**while** (itor.hasNext()) {

System.***out***.println(itor.next());

}

}

}

要求：

1. 把//1 处的代码补充完整，要求输出 list 中所有元素的内容
2. 写出程序执行的结果 Hello,Java,Learn,World
3. 如果要把实现类由 ArrayList 换为 LinkedList，应该改哪里？ArrayList 和

LinkedList 使用上有什么区别？实现上有什么区别？

ArrayList(更便于进行频繁的查询操作);

LinkedList(更便于进行频繁的增加，删除操作);

1. 如果要把实现类由 ArrayList 换为 Vector，应该改哪里？ArrayList 和 Vector 使用上有什么区别？实现上有什么区别？

List list = new Vector<>();

Vector(线程安全的,但效率相比ArrayList较低);

ArrayList（线程不安全的）；

3、（List）写出下面程序的运行结果

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Demo {

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

List list = **new** ArrayList();

list.add("Hello");

list.add("World");

list.add("Hello");

list.add("Learn");

list.remove("Hello");

list.remove(0);

**for** (**int** i = 0; i < list.size(); i++) {

System.***out***.println(list.get(i));

}

}

}

输出：Hello,Learn

4、（Set，List） **import** java.util.ArrayList;

**import** java.util.HashSet;

**import** java.util.List;

**import** java.util.Set;

**public** **class** Demo {

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

List list = **new** ArrayList();

list.add("Hello");

list.add("Learn");

list.add("Hello");

list.add("Welcome");

Set set = **new** HashSet();

set.addAll(list);

System.***out***.println(set.size());

}

}

A． 编译不通过

B． 编译通过，运行时异常

C． 编译运行都正常，输出 3

D． 编译运行都正常，输出 4

**5、（List）已知有一个 Worker 类，**如下

**package** com.psfd.util;

**public** **class** Worker {

**private** **int** age;

**private** String name;

**private** **double** salary;

**public** Worker() {

}

**public** Worker(String name, **int** age, **double** salary) {

**this**.name = name;

**this**.age = age;

**this**.salary = salary;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **double** getSalary() {

**return** salary;

}

**public** **void** setSalary(**double** salary) {

**this**.salary = salary;

}

**public** **void** work(){

System.***out***.println(name + "work");

}

}

完成下面的要求

1. 创建一个 List，在 List 中增加三个工人，基本信息如下： 姓名 年龄 工资

zhang3 18 3000

li4 25 3500

wang5 22 3200

1. 在 li4 之前插入一个工人，信息为：姓名：zhao6，年龄：24，工资 3300
2. 删除 wang5 的信息
3. 利用 for 循环遍历，打印 List 中所有工人的信息
4. 利用迭代遍历，对 List 中所有的工人调用 work 方法。
5. 为 Worker 类重写 equals 方法，当姓名、年龄、工资全部相等时候才返回 true

**public** **boolean** equals(Object obj) {

**if** (**this** == obj)

**return** **true**;

**if** (obj == **null**)

**return** **false**;

**if** (getClass() != obj.getClass())

**return** **false**;

Worker other = (Worker) obj;

**if** (age != other.age)

**return** **false**;

**if** (name == **null**) {

**if** (other.name != **null**)

**return** **false**;

} **else** **if** (!name.equals(other.name))

**return** **false**;

**if** (Double.*doubleToLongBits*(salary) != Double.*doubleToLongBits*(other.salary))

**return** **false**;

**return** **true**;

}

1. **（Set，Hash 算法）**

为上一题的 Worker 类，在添加完 equals 方法的基础上，添加一个hashCode 方法。

**public** **int** hashCode() {

// 1

}

有几种写法：

1. return 0；0;

int result = 0;

if (name != null) {

result = name.hashCode();

}

return result + age;

1. return super.hashCode();

现在要把 Worker 类放入 HashSet 中，并希望在 HashSet 中没有重复元素，则下面说法正确的是：

1. 三种写法都正确
2. 1), 2)写法正确，2)效率更高
3. 2)写法正确，1),3)写法都不正确
4. （Set，Hash 算法，方法覆盖）

**package** com.psfd.util;

**public** **class** Workers {

**private** String name;

**private** **int** age;

**private** **double** salary;

**public** Workers() {

}

**public** Workers(String name, **int** age, **double** salary) {

**this**.name = name;

**this**.age = age;

**this**.salary = salary;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**public** **double** getSalary() {

**return** salary;

}

**public** **void** setSalary(**double** salary) {

**this**.salary = salary;

}

@Override

**public** String toString() {

**return** "Workers [name=" + name + ", age=" + age + ", salary=" + salary + "]";

}

}

1. 、（Set，Hash 算法）在前面的 Worker 类基础上，为 Worker 类增加相应的

方法，使得

Worker放入 HashSet 中时，Set 中没有重复元素。并编写相应的测试代码。

【可选】

2）、（Set，Comparable 接口）在前面的 Worker 类基础上，为 Worker 类添加相应的代码， 使得 Worker 对象能正确放入 TreeSet 中。并编写相应的测试代码。

注：比较时，先比较工人年龄大小，年龄小的排在前面。如果两个工人年龄相同，则再比较其收入，收入少的排前面。如果年龄和收入都相同，则根据字典顺序比较工人姓名。例如：有三个工人，基本信息如下：

姓名 年龄 工资

zhang3 18 1500

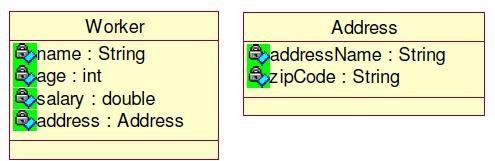
li4 18 1500

wang5 18 1600

zhao6 17 2000

放入 TreeSet 排序后结果为： zhao6 li4 zhang3 wang5

8、（Set）有如下两个类（只写了类的属性，请自行添加相应的构造方法和 get/set 方法）



要求，完善 Worker 和 Address 类，使得 Worker 对象能够正确放入 HashSet 中：即将

Worker 放入 HashSet 中时不会出现重复元素。并编写相应测试代码。

9、控制台项目：使用集合来制作学生管理系统

1）、系统启动时，创建一个学校，然后给学校一些初始化学生信息、课程信息。

2）、学生信息管理（StudentManager）

1.增加学生

2.修改学生

3.删除学生

4.查询学生

6.课程管理

7.系统退出

3）、课程信息管理(CourseManager)

1.增加课程

2.修改课程

3.删除课程

4.通过学生学号，查询该学生选课情况

5.通过课程号，打印当前课程下面学生信息

6.添加选这门课程的学生

7.删除选择这门课程的学生

8.返回上级菜单