Person

```java
public class Person {

    private String name;
    Person(String n) {
        name = n;
    }
    public String getName() {
        return name;
    }
    @Override
    public boolean equals(Object aPerson) {
        if(aPerson == null) return false;
        if(!(aPerson instanceof Person)) return false;
        Person p = (Person)aPerson;
        boolean isEqual = this.name.equals(p.name);
        return isEqual;
    }
    public static void main(String[] args) {

    }

}
```


PersonWithJob

```java
public class PersonWithJob extends Person {

    private double salary;

    public double getSalary() {
        return salary;
    }
    PersonWithJob(String n, double s) {
        super(n);
        salary = s;
    }

    @Override
    public boolean equals(Object aPerson) {
        if(aPerson == null) return false;
        if(!(aPerson instanceof PersonWithJob)) return false;
        PersonWithJob p = (PersonWithJob)aPerson;
        boolean isEqual = this.getName().equals(p.getName()) &&
                this.getSalary()==p.getSalary();
        return isEqual;
    }
    public static void main(String[] args) {
        Person p1 = new PersonWithJob("Joe", 30000);
        Person p2 = new Person("Joe");
        //As PersonsWithJobs, p1 should be equal to p2
        System.out.println("p1.equals(p2)? " + p1.equals(p2));
        System.out.println("p2.equals(p1)? " + p2.equals(p1));
    }
```

In the main method in the PersonWithJob class, two instances of Person have been compared to determine if they are equal. The comparison is done in two different ways. One way leads to a “false”, the other to a “true.” Explain why this has happened.

Answer: The `PersonWithJob` class overrides the `equals` method inherited from its parent class, `Person`. In its implementation, it checks whether the object is an instance of the `PersonWithJob` class rather than `Person`. As a result, during runtime, when the `PersonWithJob equals` method is called, `Person` object is passed to the method; the check fails, and the method returns `false`. This happens because, even though the `this` object is also of type `Person`, the passed `Person` object does not belong to the subclass `PersonWithJob`.