**Contract between equals() and hashCode() methods :**

* If two objects are equal(according to ****equals()**** method) then the ****hashCode()**** method should return the same integer value for both the objects.
* But, it is not necessary that the****hashCode()**** method will return the distinct result for the objects that are not equal (according to ****equals()**** method).

**Why do we need to Override equals and hashcode methods in Java :** Java suggests to always override hashCode() method if the class overrides equals().

1. **Override only equals() without overriding hashCode() :** Overriding only equals() method without overriding hashCode() causes the two equal instances to have unequal hash codes, that is in violation of the hashCode contract (mentioned in Javadoc) that clearly says, if two objects are equal according to the equals(Object) method, then calling the hashCode method on each of the two objects must produce the same integer result.

Since the default hashCode implementation in the Object class return distinct integers for distinct objects, if only equals() method is overridden,

**2. Override only hashCode() without overriding equals() :** If we only override hashCode() method, both e1 and e2 will hash to the same bucket as they produces the same hash code. But since equals() method is not overridden, when the set hashes e2 and iterates through the bucket looking if there is an Employee e such that e2.equals(e) is true, it won’t find any as e2.equals(e1) will be false.

Please note that even though equal objects must have equal hash codes, the reverse is not true. It is perfectly valid to override hashCode() without overriding equals() as objects with equal hash codes need not be equal.