**java.lang package – Part-02**

* **hashCode()**

For every object a unique number is generated by JVM, which is nothing but hashCode.

Hashcode won’t represent address of object.

JVM will use hashcode while saving objects into hashing related data structures like HashTable, HashMap, HashSet etc.

The main advantage of saving objects based on hashcode is, search operation will become easy. (The most powerful search algorithm up to today is hashing).

If you are giving the chance to object class hashCode() method it will generate hashcode based on address of the object. It doesn’t mean hashcode represents address of the object.

Based on our requirement we can override hashcode method in our class to generate our own hashcode.

Overriding hashcode method is said to be proper if and only if for every object we have to generate a unique number as hashcode.

class Student{

public int hashCode(){

return 100;

}

}

Improper:

This is improper way of overriding hashcode method, because for all student objects we are generating same number as hashcode.

class Student{

public int hashCode(){

return rollNo;

}

}

Proper:

This is proper way of overriding hashcode method, because we are generating a different hashcode for every object.

* **toString() VS hashCode():**

1. If we are giving the chance to Object class toString() method, it will internally calls hashCode() method.
2. If we are overriding toString() then our toString() method may not call hashcode() method.

* **Example\_01:**

class Test{

int i;

Test(int i){

this.i = i;

}

public static void main(String[] args){

Test t1 = new Test(10);

Test t1 = new Test(100);

System.out.println(t1);

System.out.println(t2);

}

}

Note:

For the above class, Object class 🡪 toString() and Object class 🡪 hashCode() methods will be called. Hence, we can’t predict the exact output.

* **Example\_02:**

class Test{

int i;

Test(int i){

this.i = i;

}

public int hashCode(){

return i;

}

public static void main(String[] args){

Test t1 = new Test(10);

Test t1 = new Test(100);

System.out.println(t1);

System.out.println(t2);

}

}

Note:

For the above class, Object class 🡪 toString() method will be called and Test class 🡪 hashCode() method will be called. So, below is the output.

Test@a

Test@64

* **Example\_03:**

class Test{

int i;

Test(int i){

this.i = i;

}

public String toString(){

return i+””;

}

public int hashCode(){

return i;

}

public static void main(String[] args){

Test t1 = new Test(10);

Test t1 = new Test(100);

System.out.println(t1);

System.out.println(t2);

}

}

Note:

The above class calls the toString() method of test class as the test class has the overridden method. Hence, it won’t call the hashCode() as only the Object class toString() method depends on hashCode() method. So, the answer is.

10

100