**Hashcode():** public int hashcode()

This method is supported for hash tables which are used in HasMaps or HashSets,it returns the hash code value for the object.

**Equals():** public boolean equals([Object](https://docs.oracle.com/javase/7/docs/api/java/lang/Object.html) obj)

This method is used in most of the collections and it is used to determine if the collections have the same element or not .

This method is reflexive, symmetric, transitive and it returns true for all these operations. For any null reference values he equals method returns false, generally to check the hash code value for the objects we have to override the has code method and this method returns true when the 2 objects have same hash code value.

Example:

**public** **class** ObjectTest {

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

// **TODO** Auto-generated method stub

String s1 = "Test";

String s2 = "Test" ;

String s3="abc";

System.***out***.println( s1.equals( s2 ));

System.***out***.println( s1.equals( s3 ));

**int** a=System.*identityHashCode*(s1);

**int** b=System.*identityHashCode*(s2);

**int** c=System.*identityHashCode*(s3);

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

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

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

}}

**ToString():**public String toString()

It returns the string representation of the object .This method can be over ridden by the subclass to give the representation of the text.The toString method is from object class which basically prints the address of the initialized object of it is not used in any class ,otherwise the method is used in a class and returned with the string representation

**Clone():**protected Object clone() throws CloneNotSupportedException

This method is used to create and return a copy of the object. Basically the same will create 2 objects and initializes one object we can copy the contents of one object to another using the clone method and is the class extends the class we can execute this method using super.clone()

**Example**:

ExampleTemp class:

**public** **class** ExampleTemp {

**private** **int** id;

**private** String name;

**private** **float** price;

//private Bid[] bidsArr;

**public** **int** getId() {

**return** id;

}

**public** ExampleTemp(**int** id, String name, **float** price) {

**this**.id = id;

**this**.name = name;

**this**.price = price;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

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

**this**.name = name;

}

**public** **float** getPrice() {

**return** price;

}

**public** **void** setPrice(**float** price) {

**this**.price = price;

}

@Override

**public** String toString()

{

String temp = "ID: "+**this**.getId()+" Name: "+**this**.getName()+" Price: "+**this**.getPrice();

**return** temp;

}

**protected** Object getClone() {

**try** {

**return** **this**.clone();

} **catch** (CloneNotSupportedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**return** **null**;

}

}

Test Class:

**public** **class** ObjectTest {

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

// **TODO** Auto-generated method stub

ObjectTest o=**new** ObjectTest();

ExampleTemp e=**new** ExampleTemp(77,"bb",99);

ExampleTemp e1 = e;

System.***out***.println("First object contents "+e);

System.***out***.println("second object contents "+e1);

}

}

Output:

First object contents ID: 77 Name: bb Price: 99.0

second object contents ID: 77 Name: bb Price: 99.0

**getclass():**

Determines the runtime class of this Object

**Finalize():**protected void finalize() throws Throwable

This method is called by the garbage collector when the garbage collection determines that there are no more references to the object .When the class extends the object class it basically over rides the finalize() method to perform cleanup of system resources .This do not return any value.

protected void finalize() throws Throwable {

try {

close();

} finally {

super.finalize();

}

}

The above code closes the files.