**Day-1 Java Assignment**

**1. Problem Description:**

What is the difference between equals() and == in Java?

Tell the output of the given code.

public class Dog {}

public class Test {

public static void main(String[] args) {

int n1 = 10;

int n2 = 10;

System.out.println(n1 == n2);

boolean b1 = true;

boolean b2 = false;

System.out.println(b1 == b2);

Dog dog1 = new Dog();

Dog dog2 = new Dog();

System.out.println(dog1 == dog2);

System.out.println(dog1.equals(dog2));

String s1 = "Hello! World!";

String s2 = "Hello! World!";

String s3 = new String("Hello! World!");

System.out.println(s1 == s2);

System.out.println(s1 == s3);

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

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

}

}

**2. My Solution:**

* equals() is a method and == is an operator in Java and both are used to compare strings.
* The equals() method compares the actual contents of two objects. It is defined in the Object class.
* The == Operator compares the object references to see if they refer to the same object in memory.

**Answers to given code:**

public class Dog {}

public class Test {

public static void main(String[] args) {

int n1 = 10;

int n2 = 10;

System.out.println(n1 == n2); //True, we are checking primitive type values using the "==" operator, so internally it will compare the values of the variables.

boolean b1 = true;

boolean b2 = false;

System.out.println(b1 == b2); //False, internally, the == operator checks if the values of the Boolean variables are equal.

Dog dog1 = new Dog();

Dog dog2 = new Dog();

System.out.println(dog1 == dog2); //False, using == operator, we are comparing the memory addresses or references of the objects, not their actual content.

System.out.println(dog1.equals(dog2)); //False, it compares object references (memory addresses) by default. Since both abject has different references the resultant is False.

String s1 = "Hello! World!";

String s2 = "Hello! World!";

String s3 = new String("Hello! World!");

System.out.println(s1 == s2); //True, both s1 and s2 refer to the same string literal in the string pool, that’s why the result is True

System.out.println(s1 == s3); //False, In this example memory reference is different, but the value is those diff. memory is same.

System.out.println(s1.equals(s2)); //True, the equals() method compares the content of the strings s1 and s2. Since both s1 and s2 are string literals with the same content "Hello! World!", the comparison evaluating true.

System.out.println(s1.equals(s3)); //True, here equals() method compares the content of the strings s1 and s3. Despite being created differently (one is a string literal, the other is a new String object), both s1 and s3 contain the same sequence of characters "Hello! World!".

}

}