**Java-8 – Predefined functional interfaces – Predicate – Part\_02**

* **Predicate example:**

public boolean test(Integer I){

if(I > 10)

return true;

else

return false;

}

Note: A lambda expression is a method without name, return type and access modifier. So remove all things from the above method, then it will become lambda expression.

(Integer I) -> {

If(I > 10)

return true;

else

return false;

};

We can rewrite the above lambda expression as below:

I -> I > 10;

Because JVM can predict the type at runtime.

Using Predicate:

Functional interfaces can be used to refer the lambda expression.

Predicate<Integer> predicate = I -> I > 10;

System.out.println(predicate.test(100)); // true

* **Example:**

import java.util.function.Predicate;

class PredicateDemo{

Predicate<Integer> predicate = I -> I > 10;

System.out.println(predicate.test(100)); // true

System.out.println(predicate.test(5)); // false

System.out.println(predicate.test(“durga”)); //CE

Incompatible type, string cannot be converted to integer.

}