# 16.DecimaltoBinaryandOctalConversion(White-Box Testing)

Aim:

TowriteaJavaprogramthatconvertsadecimalnumberintoitsbinaryandoctalequivalents and validate the output using white-box testing with JUnit.

Algorithm:

1. **Step1:**Accept thedecimal numberasinput.
2. **Step2:**Convert thedecimalnumberintobinaryusingJava'sbuilt-in functionality.
3. **Step3:**ConvertthedecimalnumberintooctalusingJava'sbuilt-in functionality.
4. **Step4:**Displaythe binaryandoctal equivalents.
5. **Step5:**WriteJUnittestcasestoverifythecorrectnessoftheconversionlogic, ensuring that both binary and octal conversions are accurate.

**Code:**

**Step 1:**

public class NumberConverter {

public String toBinary(int number) {

if (number >= 0) {

return Integer.toBinaryString(number);

} else {

return "-" + Integer.toBinaryString(-number);

}

}

public String toOctal(int number) {

if (number >= 0) {

return Integer.toOctalString(number);

} else {

return "-" + Integer.toOctalString(-number);

}

}

}

**Step 2:**

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class NumberConverterTest {

private NumberConverter converter;

@BeforeEach

public void setUp() {

converter = new NumberConverter();

}

@Test

public void testConvert10() {

assertEquals("1010", converter.toBinary(10)); // Binary

assertEquals("12", converter.toOctal(10)); // Octal

}

@Test

public void testConvert0() {

assertEquals("0", converter.toBinary(0)); // Binary

assertEquals("0", converter.toOctal(0)); // Octal

}

@Test

public void testConvert255() {

assertEquals("11111111", converter.toBinary(255)); // Binary

assertEquals("377", converter.toOctal(255)); // Octal

}

@Test

public void testConvertNegative10() {

assertEquals("-1010", converter.toBinary(-10)); // Binary

assertEquals("-12", converter.toOctal(-10)); // Octal

}

}

Sample Input:

# TestCase1:

* + - Input: 10
    - ExpectedOutput:
      * Binary:1010
      * Octal:12

# TestCase2:

* + - Input:0
    - ExpectedOutput:
      * Binary:0
      * Octal:0

# TestCase3:

* + - Input: 255
    - ExpectedOutput:
      * Binary:11111111
      * Octal:377

# TestCase4:

* + - Input:-10
    - ExpectedOutput:
      * Binary:-1010
      * Octal:-12

SampleOutput:

# TestCase1:

* + - Input: 10
    - Output:
      * Binary:1010
      * Octal:12

# TestCase2:

* + - Input:0
    - Output:
      * Binary:0
      * Octal:0

# TestCase3:

* + - Input: 255
    - Output:
      * Binary:11111111
      * Octal:377

# TestCase4:

* + - Input:-10
    - Output:
      * Binary:-1010
      * Octal:-12

Results:

* + **TestCase1:**Theprogramcorrectlyconverts10 tobinary1010 andoctal 12.
  + **TestCase2:**Theprogramcorrectlyhandlesthecasewheretheinputis0,outputting both binary and octal as 0.
  + **TestCase3:**Theprogramcorrectlyconverts255to binary11111111andoctal 377.
  + **TestCase4:**Theprogramcorrectlyhandlesthecaseofanegativenumber-10, converting it to binary -1010 and octal -12.



