1. **Check if a number is even or odd using bitwise operators:**

```
public class EvenOddCheck {
    public static void main(String[] args) {
        int num = 12;
        if ((num & 1) == 0) {
            System.out.println(num + " is even.");
        } else {
            System.out.println(num + " is odd.");
        }
    }
}
```

2. **Swap two numbers without using a temporary variable:**

```
public class SwapNumbers {
    public static void main(String[] args) {
        int a = 5, b = 10;
        a = a ^ b;
        b = a ^ b;
        a = a ^ b;
        System.out.println("After swapping: a = " + a + ", b = " + b);
    }
}
```

3. **Check if a given bit is set or not:**

```
public class CheckBit {
    public static void main(String[] args) {
```

```
        int num = 8;
        int bitPosition = 3;
        if ((num & (1 << bitPosition)) != 0) {
            System.out.println("Bit at position " + bitPosition + " is set.");
        } else {
            System.out.println("Bit at position " + bitPosition + " is not set.");
        }
    }
}
```

4. **Toggle a particular bit in a number:**

```
public class ToggleBit {
    public static void main(String[] args) {
        int num = 9;
        int bitPosition = 2;
        num = num ^ (1 << bitPosition);
        System.out.println("After toggling bit at position " + bitPosition + ": " + num);
    }
}
```

5. **Count set bits (1s) in a binary representation of a number:**

```
public class CountSetBits {
    public static void main(String[] args) {
        int num = 25;
        int count = 0;
        while (num > 0) {
            count += num & 1;
```

```
            num >>= 1;
        }
        System.out.println("Number of set bits: " + count);
    }
}
```

6. **Check if a number is a power of 2:**

```
public class PowerOfTwo {
    public static void main(String[] args) {
        int num = 16;
        if ((num & (num - 1)) == 0) {
            System.out.println(num + " is a power of 2.");
        } else {
            System.out.println(num + " is not a power of 2.");
        }
    }
}
```

7. **Find the single non-repeating element in an array:**

```
public class SingleNonRepeating {
    public static void main(String[] args) {
        int[] arr = { 2, 4, 6, 8, 4, 6, 2 };
        int result = 0;
        for (int num : arr) {
            result ^= num;
        }
        System.out.println("Single non-repeating element: " + result);
```

```
    }
}
```

8. **Set the rightmost unset bit:**

```
public class SetRightmostUnsetBit {
    public static void main(String[] args) {
        int num = 10;
        int result = num | (num + 1);
        System.out.println("After setting the rightmost unset bit: " + result);
    }
}
```

9. **Check if a number is a palindrome in binary representation:**

```
public class BinaryPalindrome {
    public static void main(String[] args) {
        int num = 9;
        int reversed = Integer.reverse(num);
        if (num == reversed) {
            System.out.println(num + " is a binary palindrome.");
        } else {
            System.out.println(num + " is not a binary palindrome.");
        }
    }
}
```

0. **Rotate bits of a number to the left:**

```java
public class RotateBitsLeft {
    public static void main(String[] args) {
        int num = 5;
        int numBits = 4;
        int result = (num << numBits) | (num >>> (32 - numBits));
        System.out.println("After rotating bits to the left: " + result);
    }
}
```