Power of Two

Check if a number is a power of two using bit manipulation.

Power of Two (Bit Manipulation)

Problem

Given an integer **n**, check whether it is a **power of 2**.

A number is a power of two if:

```
1, 2, 4, 8, 16, 32, ...
```

These numbers have **only ONE '1'** in their binary form.

Key Bit Trick

Important Property

For any power of 2:

```
n & (n - 1) == 0
```

Why this works?

Example:

```
n = 8  → 1000
n-1=7  → 0111

1000
0111
-----
0000
```

Because a power of 2 has only one set bit.

Algorithm Steps

1. Number must be **positive**
2. Check:

```
(n & (n - 1)) == 0
```

Java Code

```
public class PowerOfTwo {

    public static boolean isPowerOfTwo(int n) {
        // Step 1: number must be positive
        if (n <= 0) return false;

        // Step 2: bit manipulation trick
        return (n & (n - 1)) == 0;
    }

    public static void main(String[] args) {
        int n = 16;

        if (isPowerOfTwo(n))
            System.out.println(n + " is a Power of Two");
        else
```

```
            System.out.println(n + " is NOT a Power of Two");
        }
    }
```