Bitwise OR

The **OR** bitwise operation takes two sets of bits and for each pair of bits (the two bits at the same index in each set) returns 1 if **either** of the bits are 1. Otherwise, it returns 0.

```java
1 | 1 // 1
1 | 0 // 1
0 | 1 // 1
0 | 0 // 0
```

Think of it like a bucket with two holes in it. If *both* holes are closed, no water comes out. If *either* hole is open, *or if both* are open, water comes out.

When performing OR on two integers, all digit columns used by either of the integers remain:

Java ▾

```
5 | 6 // Gives 7


// At the bit level:
//     0101  (5)
//   | 0110  (6)
//   = 0111  (7)
```

See also:

- Binary Numbers (/concept/binary-numbers)
- Bitwise AND (/concept/and)
- Bitwise NOT (/concept/not)
- Bitwise XOR (eXclusive OR) (/concept/xor)
- Bit Shifting (/concept/bit-shift)

What's next?

If you're ready to start applying these concepts to some problems, check out our mock coding interview questions (/next).

They mimic a real interview by offering hints when you're stuck or you're missing an optimization.

Try some questions now ➡