Brian Kernighan's Algorithm — Interview & Coding Notes (C++)

What it does:
Counts the number of set bits (1s) in an integer faster than checking every bit.

Core Trick:
n = n & (n - 1)
This removes the rightmost set bit from n.

Why it works:
Subtracting 1 from a number flips all bits after the last set bit. AND operation with the original number removes that last 1 bit.

C++ Implementation:

```
int countSetBits(int n) {
  int cnt = 0;
  while (n > 0) {
    n = n & (n - 1);
    cnt++;
  }
  return cnt;
}
```

Time Complexity:
Let k be the number of set bits.
Time = O(k)
Worst Case = O(log n)
Space = O(1)

Example:
n = 12 (1100)
12 -> 8 -> 0
Set Bits = 2

Power of 2 Check:

```
bool isPowerOfTwo(int n) {
  return n > 0 && (n & (n - 1)) == 0;
}
```

Interview Summary:
Brian Kernighan's algorithm efficiently counts set bits by removing the lowest set bit in each iteration with O(k) time and O(1) space.