When Should We Stop Recursing?

What's the problem?

In math,
- we used base cases
- to stop the recursion.

In a **C** function,
- we **need a stopping condition**
- to stop the recursion.

So: when should we stop?

Stop if the Space Has Already Been Marked as Reachable

Stop if we already reached (x,y).

```
void can_reach (int x, int y)
{
    if (found[x][y]) { return; }
    found[x][y] = 1;
    if (0 == (maze[x][y] & 1)) {
        can_reach (x - 1, y);
    }
```


Review: Induction, Bit-Slicing, Recursion

As mentioned in 120, the following are closely related mathematically
- proof by induction
- bit-sliced hardware design
- recursion.

In all three, one
- solves a small piece of a problem, then
- combines it with the "rest" of the solution
- (which is also solved as small pieces).

A General Strategy for Recursion

Here's a general strategy for recursion.

```
______ recursive ( ______ )
{
    // Check stopping conditions.
    // Handle one node.
    // Handle children.
}
```

These may be swapped.

