Ready to Write the Recursive Function

Now we're ready to write the function.

Here's a signature:

```
void can_reach (int x, int y);
```

The function should

- **set all locations reachable from `(x,y)` to 1 in `found`**, and
- **set `saw_exit` to 1 iff the exit is reachable from `(x,y)`.**

(To do so, the function will call itself.)

Mark as Reachable, then Check Children

```
void can_reach (int x, int y)
{
    found[x][y] = 1;
    if (0 == (maze[x][y] & 1)) {
        can_reach (x - 1, y);
    }
    if ((0 == maze[x][y] & 2)) {
        can_reach (x + 1, y);
    }
}
```

`(x,y)` is reachable.

No left wall?

Space to left is reachable.

Same Check and Marking for Right Child (Value 2)

```
void can_reach (int x, int y)
{
    found[x][y] = 1;
    if (0 == (maze[x][y] & 1)) {
        can_reach (x - 1, y);
    }
    if (0 == (maze[x][y] & 2)) {
        can_reach (x + 1, y);
    }
}
```

No right wall?

Space to right is reachable.

Same Check and Marking for Upper Child (Value 4)

```
    if (0 == (maze[x][y] & 4)) {
        can_reach (x, y - 1);
    }
    if (0 == (maze[x][y] & 8)) {
        can_reach (x, y + 1);
    }
    if (0 != (maze[x][y] & 16)) {
        saw_exit = 1;
    }
}
```

No upper wall?

Space above is reachable.

