The fusc function is defined recursively as follows:

```
1. fusc(0) = 0
2. fusc(1) = 1
3. fusc(2 * n) = fusc(n)
4. fusc(2 * n + 1) = fusc(n) + fusc(n + 1)
```

The 4 rules above are sufficient to determine the value of `fusc` for any non-negative input `n`. For example, let's say you want to compute `fusc(10)`.

1. `fusc(10) = fusc(5)`, by rule 3.
2. `fusc(5) = fusc(2) + fusc(3)`, by rule 4.
3. `fusc(2) = fusc(1)`, by rule 3.
4. `fusc(1) = 1`, by rule 2.
5. `fusc(3) = fusc(1) + fusc(2)` by rule 4.
6. `fusc(1)` and `fusc(2)` have already been computed are both equal to `1`.

Putting these results together `fusc(10) = fusc(5) = fusc(2) + fusc(3) = 1 + 2 = 3`

Your job is to produce the code for the `fusc` function. In this kata, your function will be tested with small values of `n`, so you should not need to be concerned about stack overflow or timeouts.

Hint: Use recursion.

When done, move on to Part 2.