61A Extra Lecture 2

Monday, September 14

Announcements

Announcements

You have a midterm in two days.

Decision Making

The Hog End Game

The Hog End Game

You: 98
Them: 99

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

S_n - Score from rolling **n** dice

t - A single outcome of rolling once

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

S_n - Score from rolling **n** dice

t - A single outcome of rolling once

$$P(S_n \geq k) = \sum_{t=2}^{6} P(t) \cdot P(S_{n-1} \geq k - t)$$

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

S_n - Score from rolling **n** dice

t - A single outcome of rolling once

$$P(S_n \geq k) = \sum_{t=2}^{6} P(t) \cdot P(S_{n-1} \geq k - t)$$

1 die n-1 dice

The Hog End Game

You: 98 **Them:** 99	**You:** 92 **Them:** 99	**You:** 88 **Them:** 99	**You:** 80 **Them:** 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

S_n - Score from rolling **n** dice

t - A single outcome of rolling once

$$P(S_n \geq k) = \sum_{t=2}^{6} P(t) \cdot P(S_{n-1} \geq k - t)$$

1 die n-1 dice

Sum over each possible dice outcome t that does not Pig Out:

(The chance to roll **t**) * (The chance to roll at least **k - t** points using **n - 1** rolls)

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

S_n - Score from rolling **n** dice

t - A single outcome of rolling once

$$P(S_n \geq k) = \sum_{t=2}^{6} P(t) \cdot P(S_{n-1} \geq k - t)$$

1 die n-1 dice

Sum over each possible dice outcome t that does not Pig Out:
(The chance to roll **t**) * (The chance to roll at least **k - t** points using **n - 1** rolls)

Base case #1: The chance to score 0 (or less) in 0 rolls is 1 (guaranteed)

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

S_n - Score from rolling **n** dice

t - A single outcome of rolling once

$$P(S_n \geq k) = \sum_{t=2}^{6} P(t) \cdot P(S_{n-1} \geq k - t)$$

1 die n-1 dice

Sum over each possible dice outcome t that does not Pig Out:
(The chance to roll **t**) * (The chance to roll at least **k - t** points using **n - 1** rolls)

Base case #1: The chance to score 0 (or less) in 0 rolls is 1 (guaranteed)

Base case #2: The chance to score more than 0 in 0 rolls is 0 (impossible)

The Hog End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

What is the chance I'll score **k** points rolling **n** 6-sided dice?

S_n - Score from rolling **n** dice

t - A single outcome of rolling once

$$P(S_n \geq k) = \sum_{t=2}^{6} P(t) \cdot P(S_{n-1} \geq k - t)$$

1 die n-1 dice

Sum over each possible dice outcome t that does not Pig Out:
(The chance to roll **t**) * (The chance to roll at least **k - t** points using **n - 1** rolls)

Base case #1: The chance to score 0 (or less) in 0 rolls is 1 (guaranteed)

Base case #2: The chance to score more than 0 in 0 rolls is 0 (impossible)

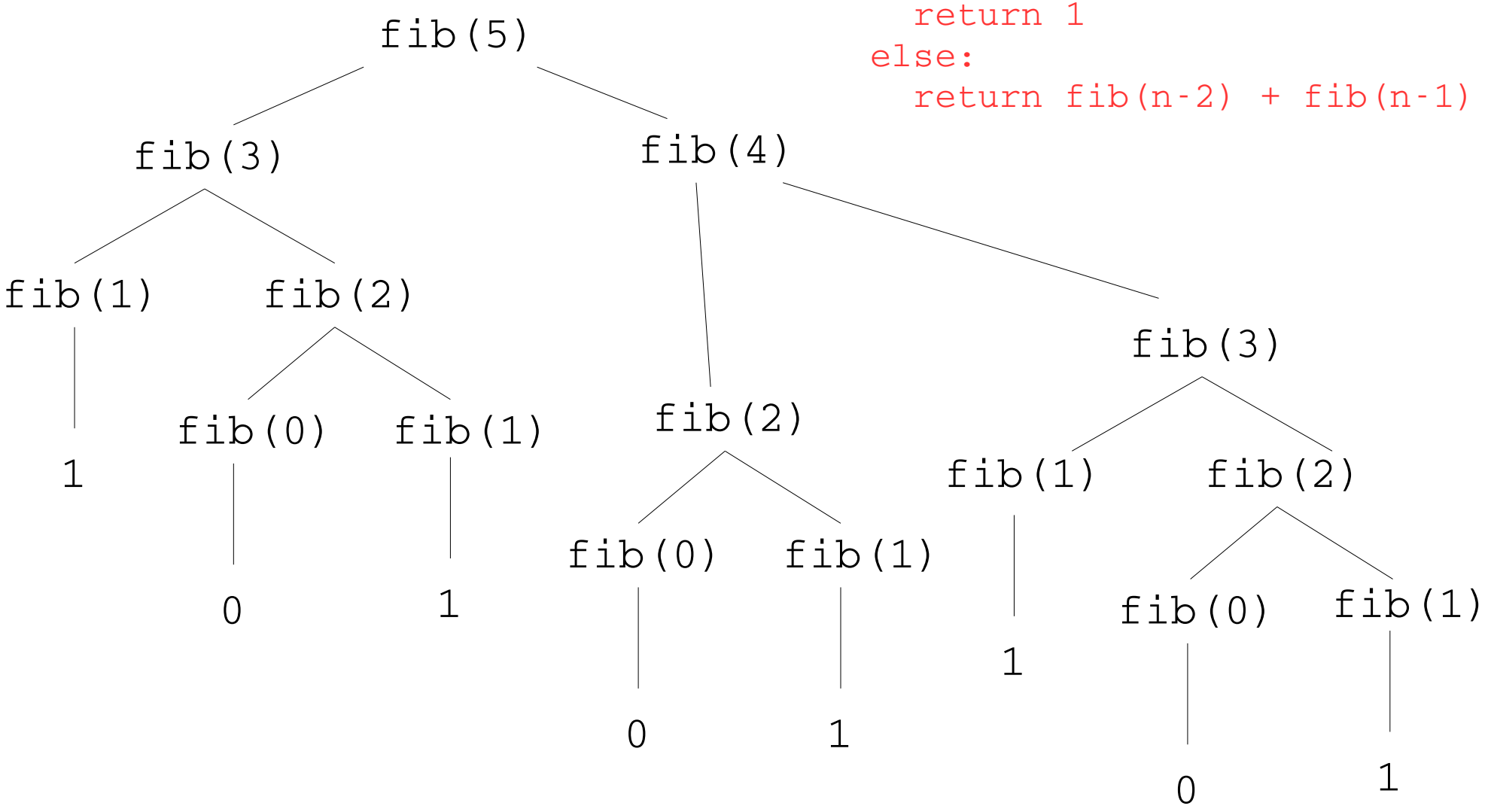
Can compute using tree recursion!!!

Memoization

(Sort of like memorization)

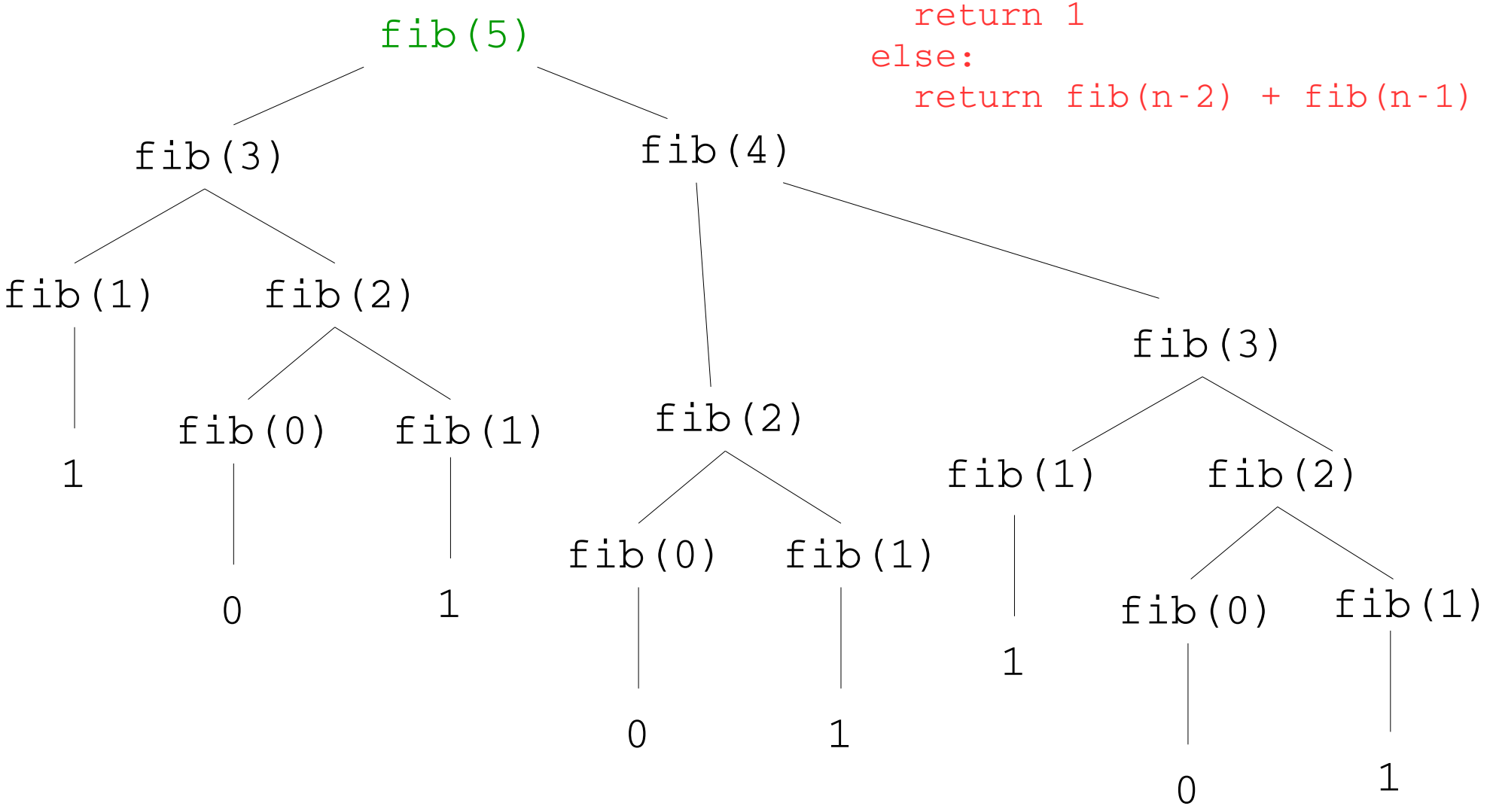
Computing Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



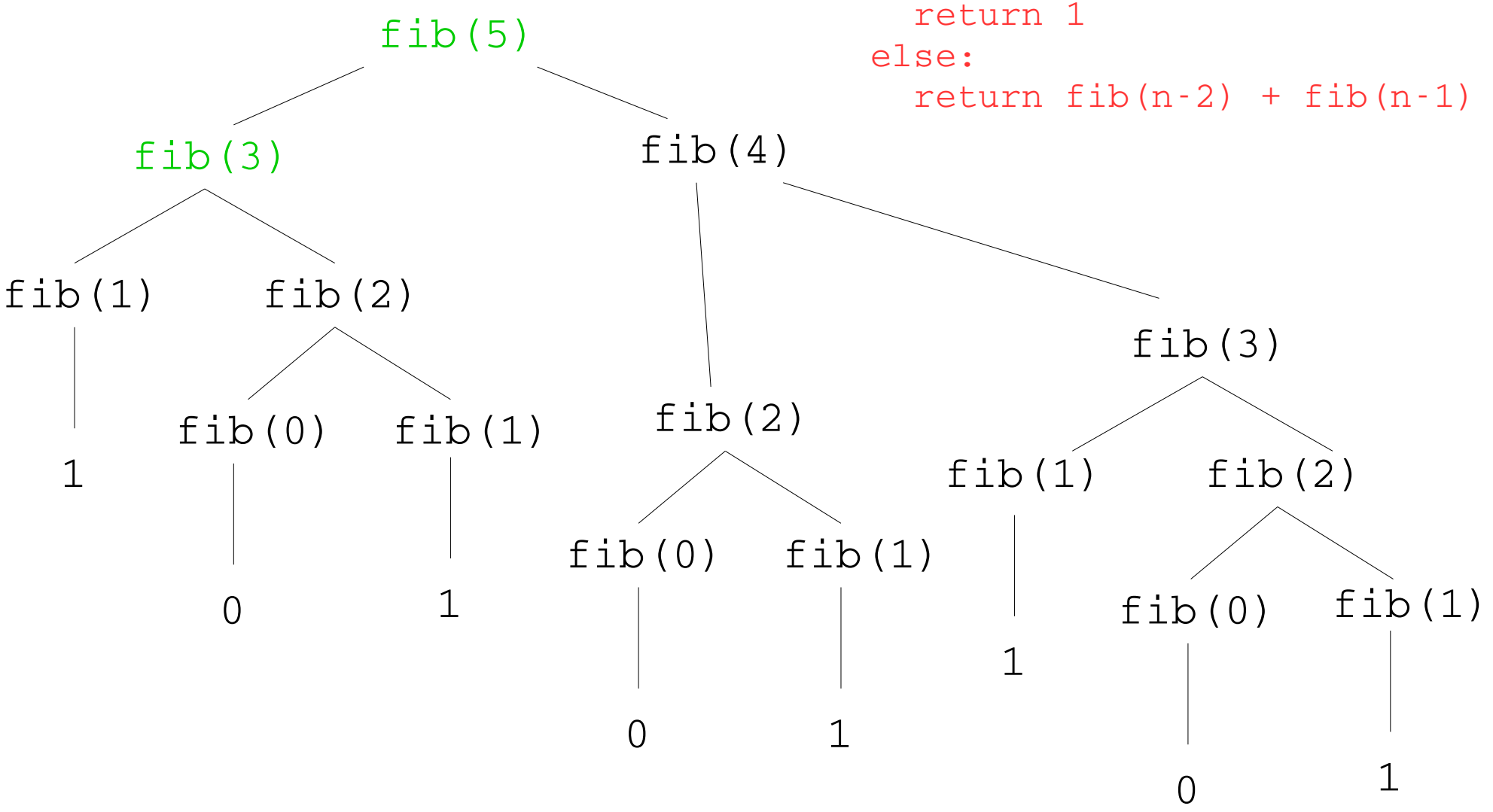
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



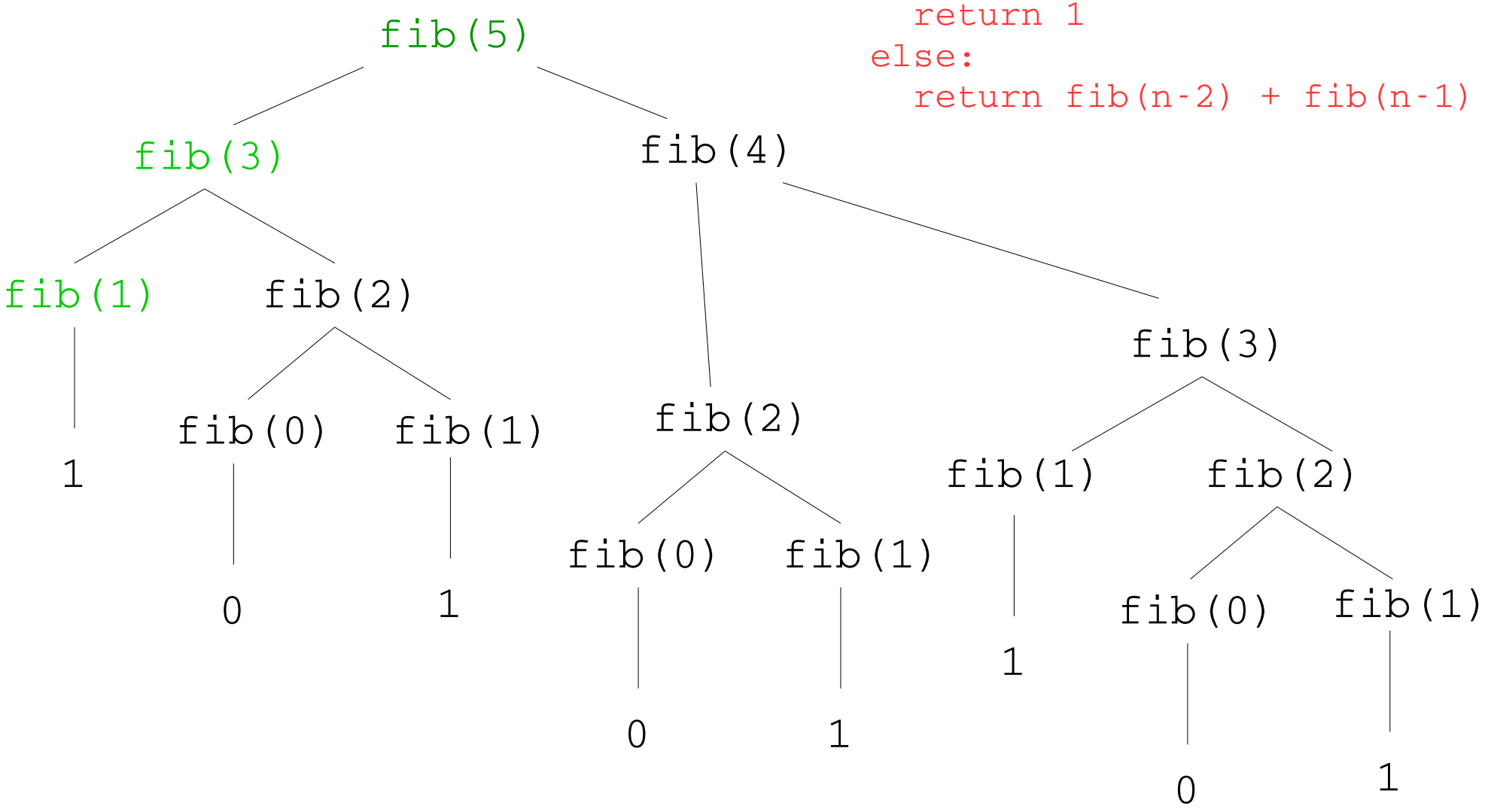
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



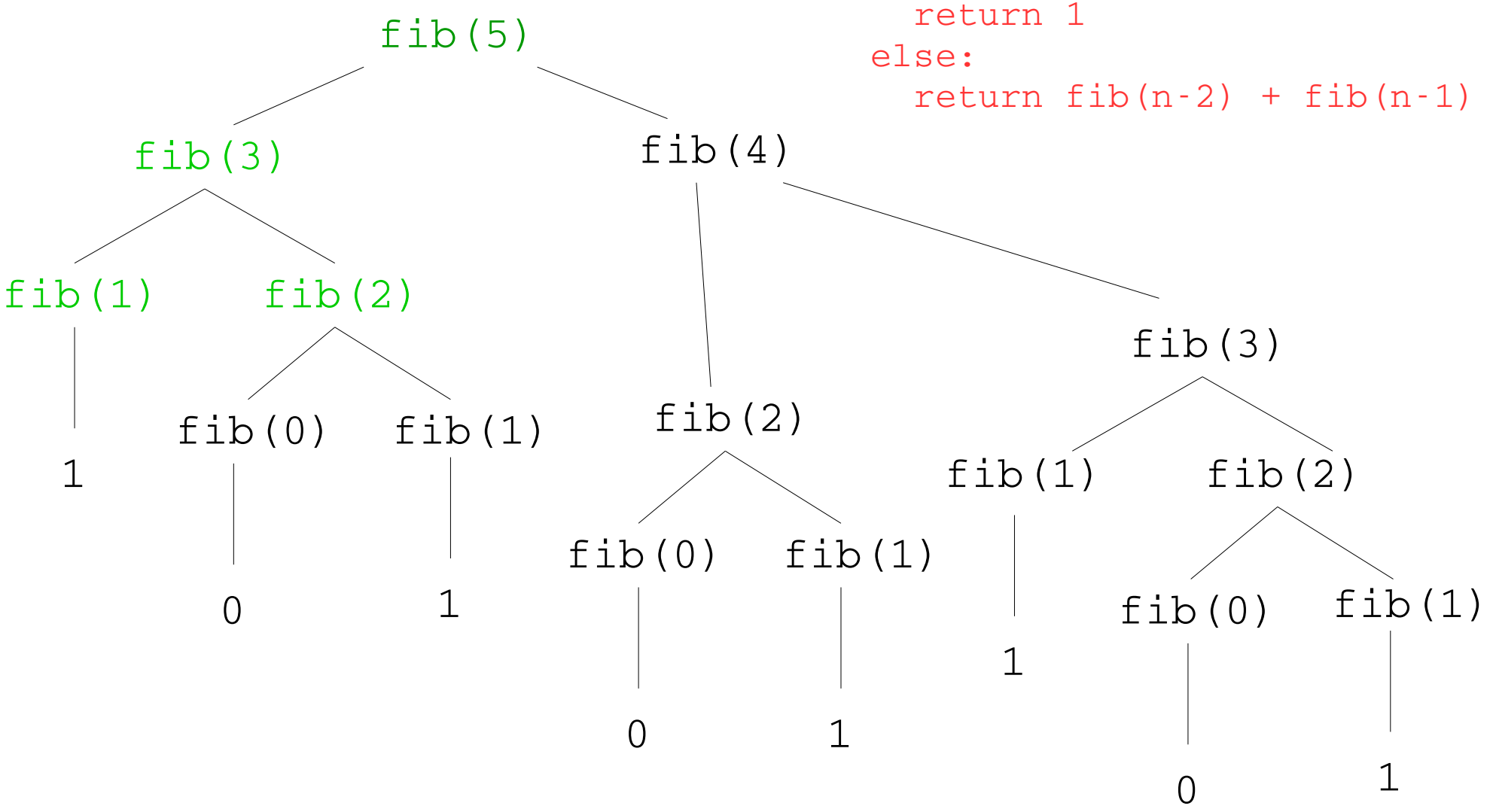
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



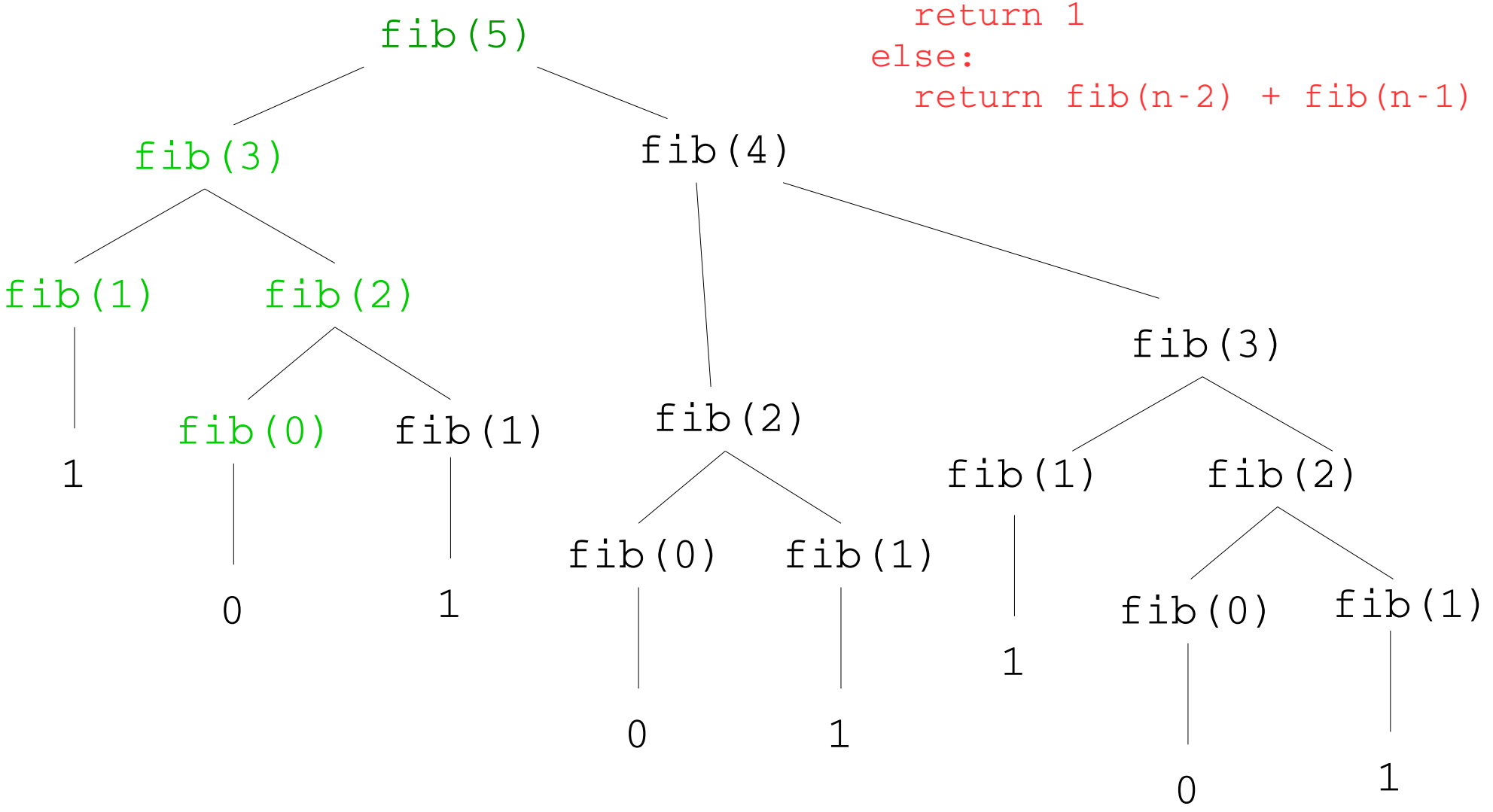
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



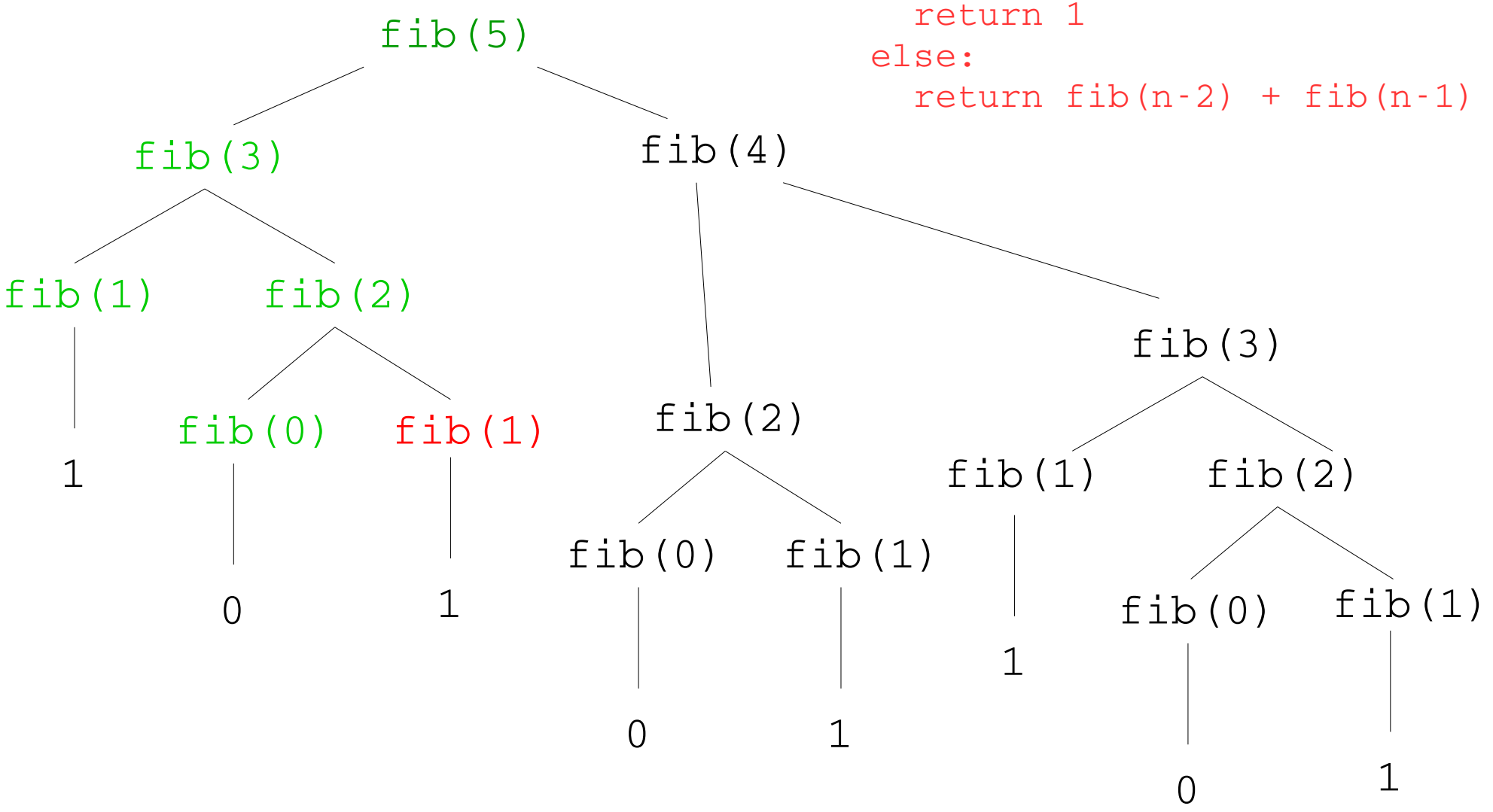
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



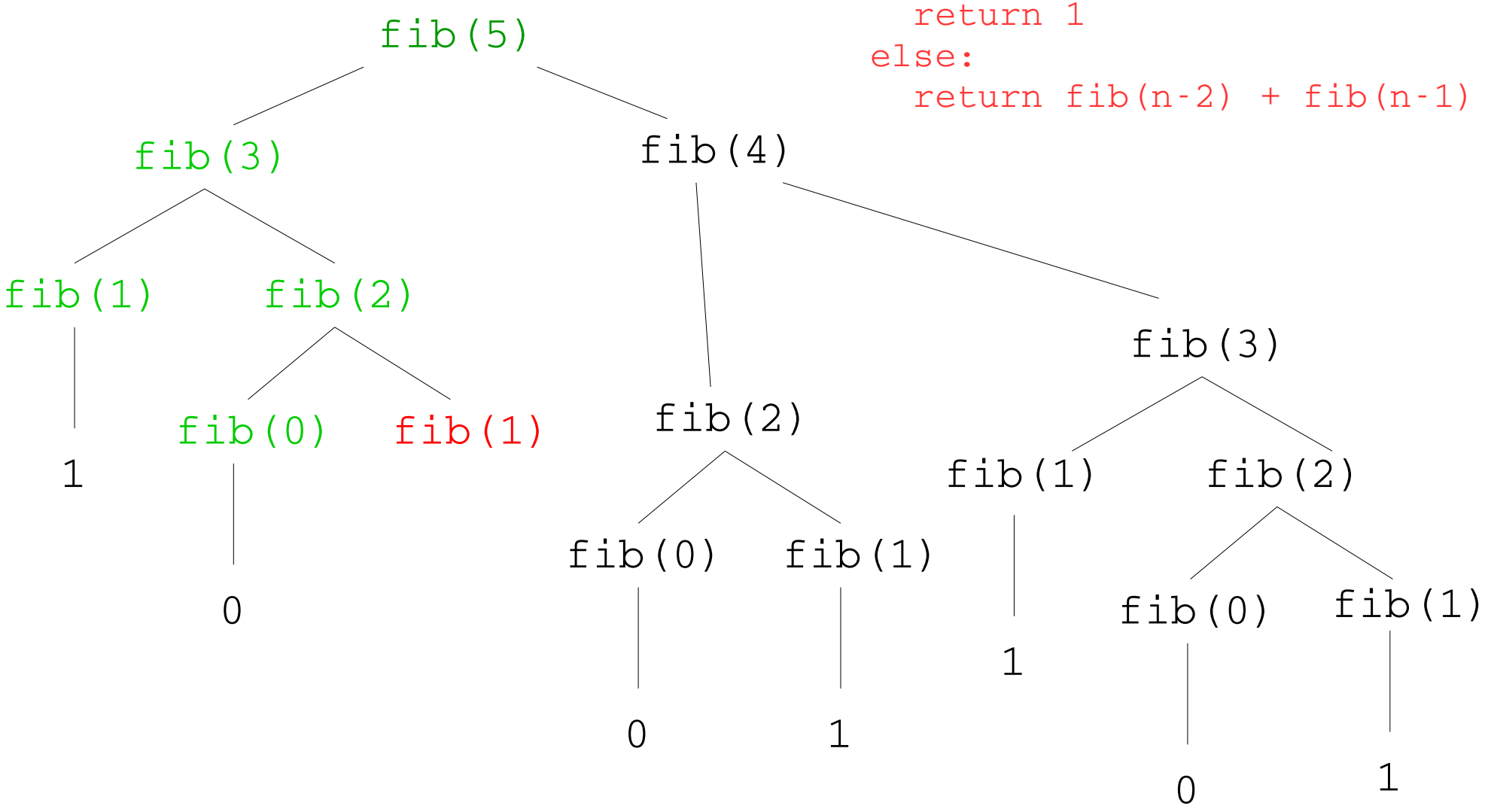
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



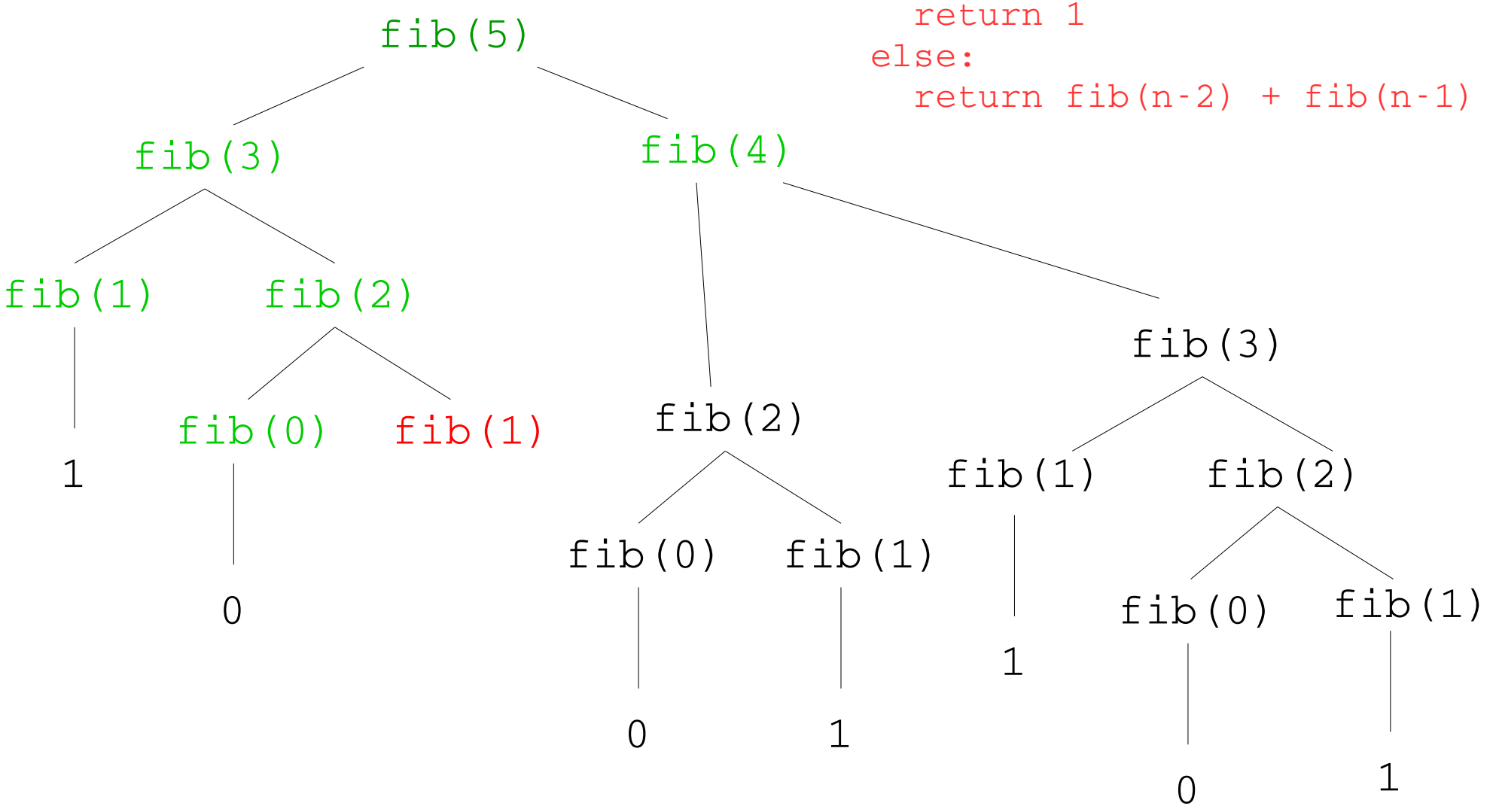
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



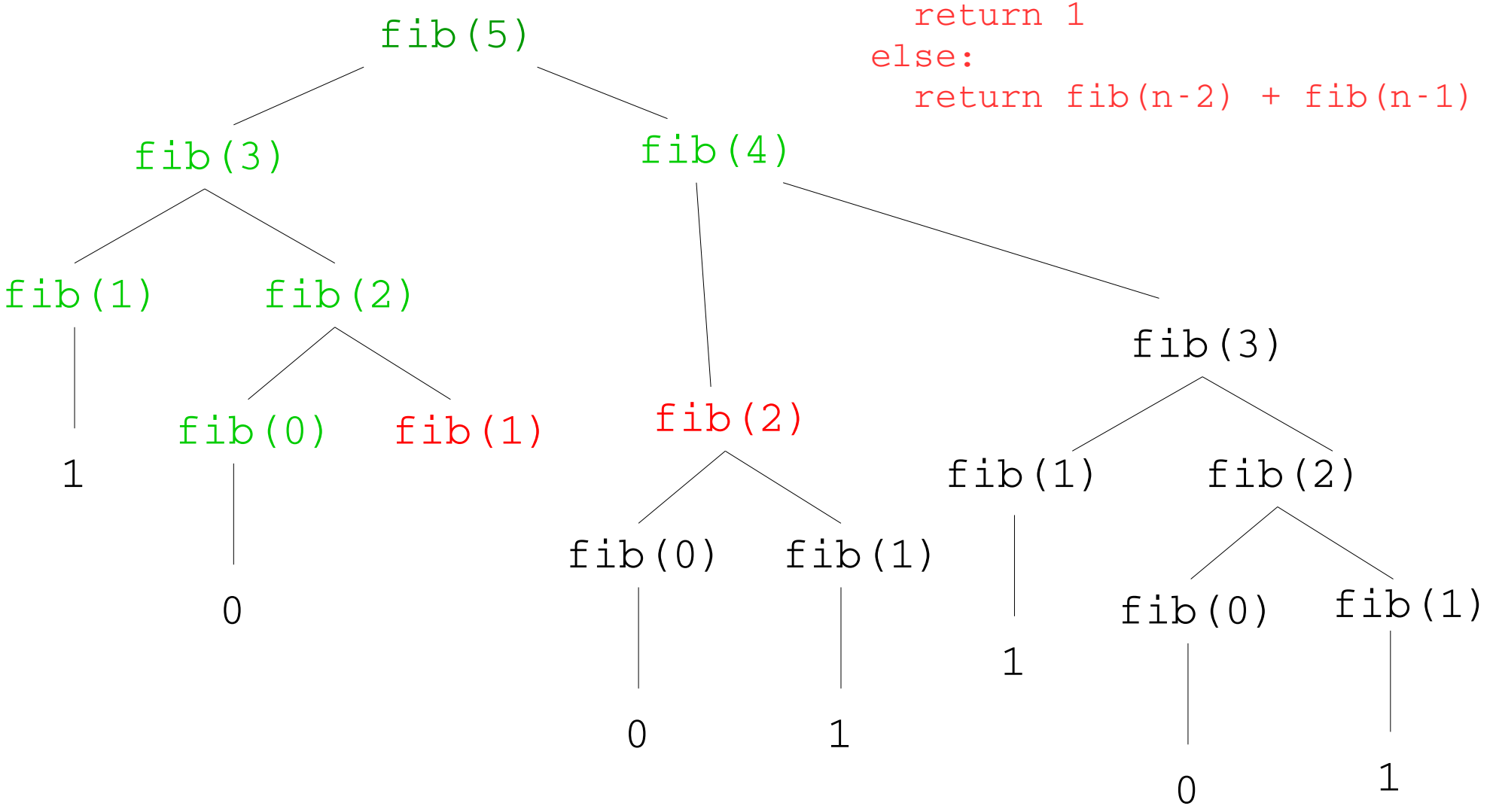
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



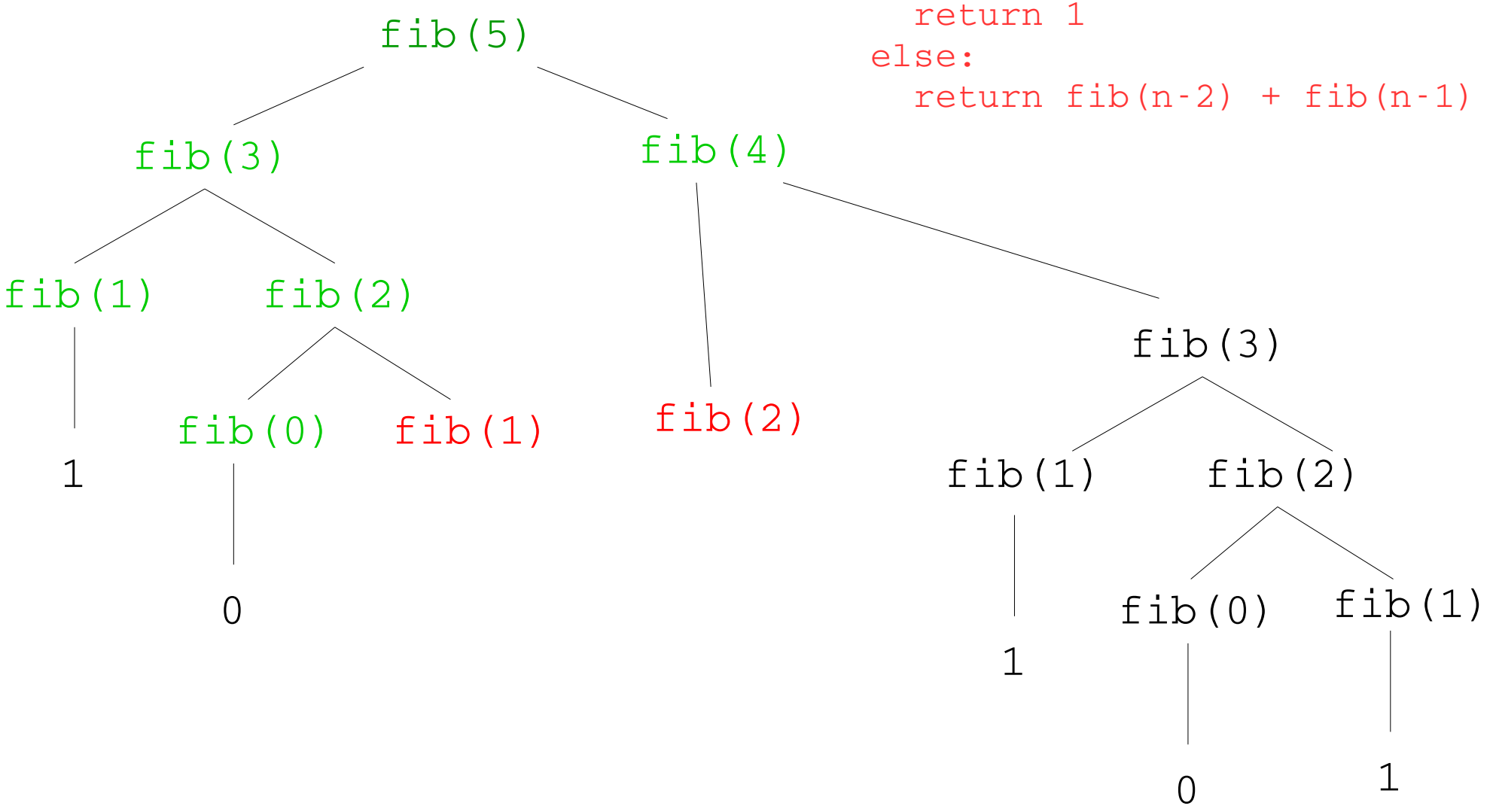
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



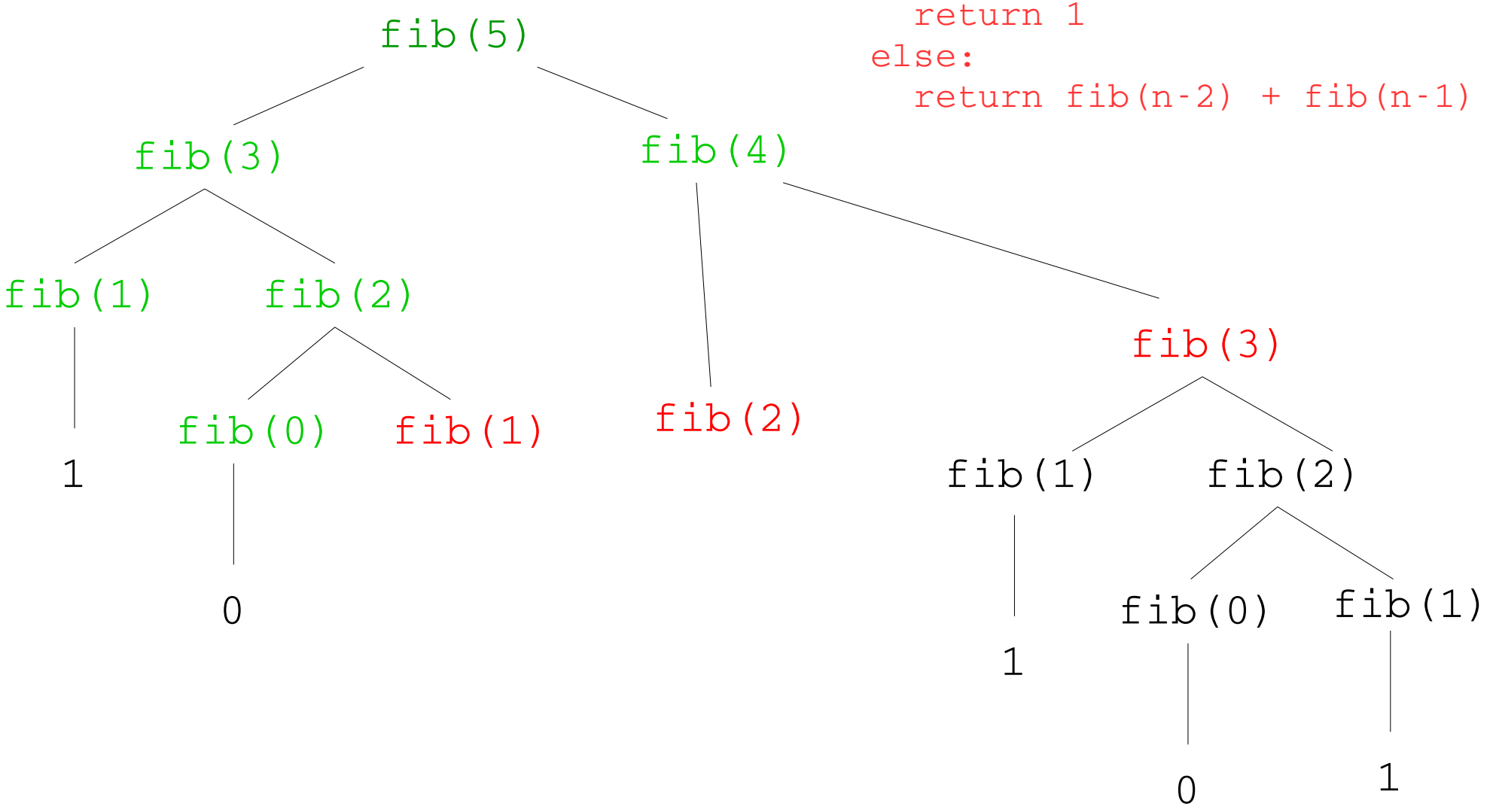
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



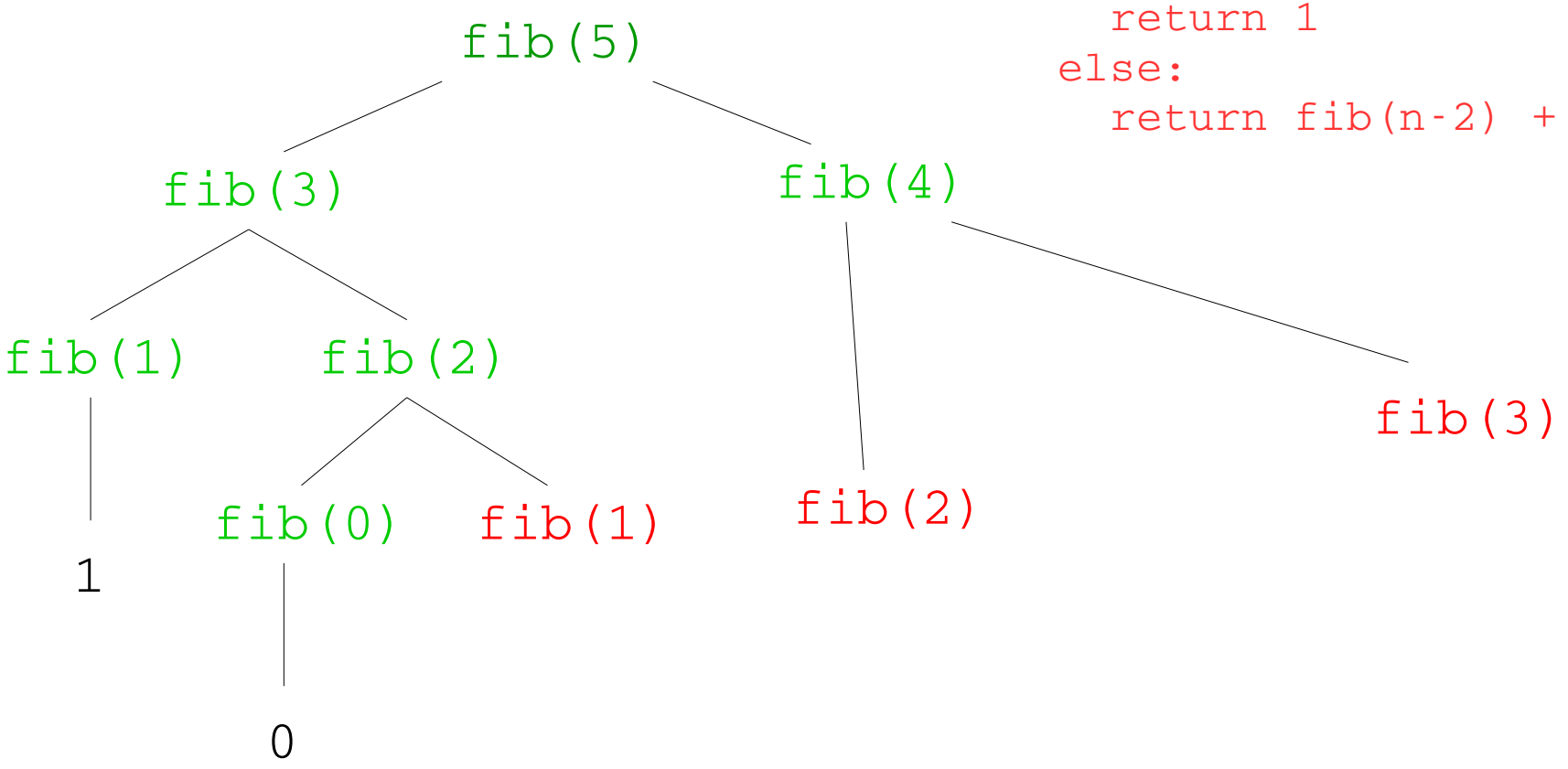
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



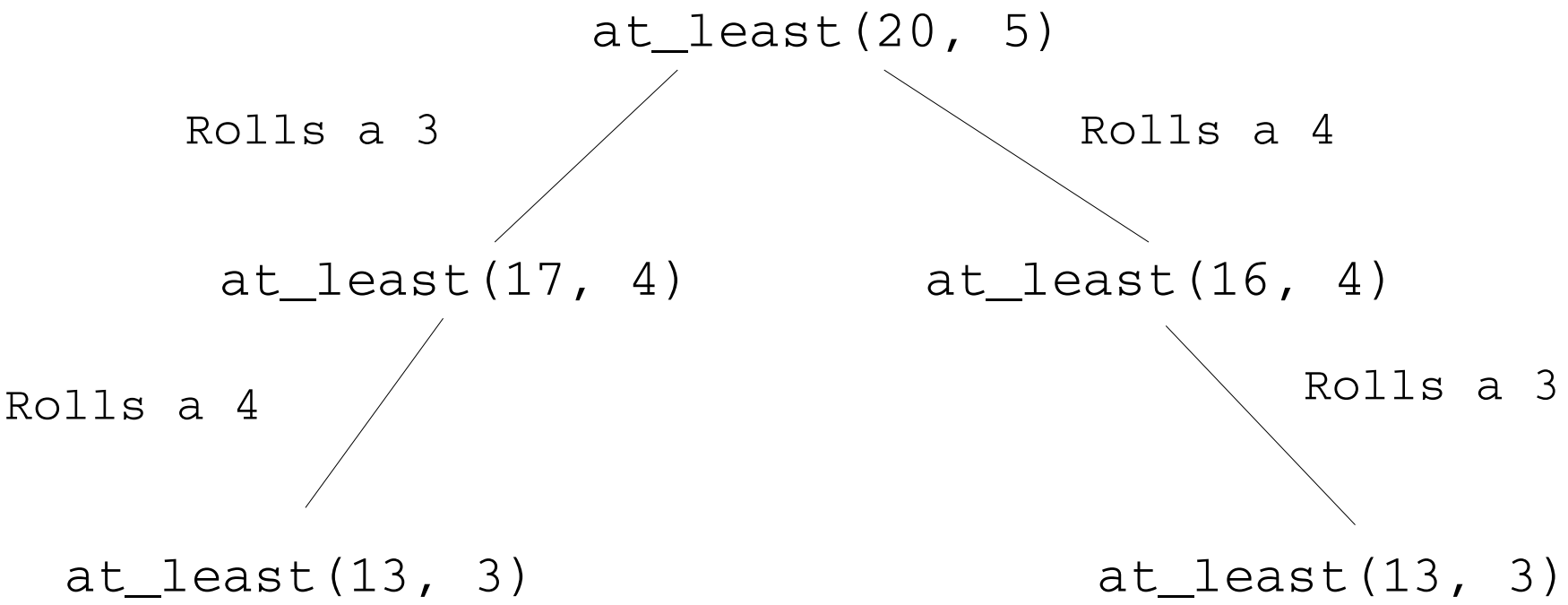
Memoized Fibonacci

```
def fib(n):
  if n == 0:
    Return 0
  elif n == 1:
    return 1
  else:
    return fib(n-2) + fib(n-1)
```



Memoized at_least

- Rolling a 3 then 4 is equivalent to 4 then 3



- We can just use the same calculated value again!

Twenty-One (Nim)

Twenty-One Rules

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more
- The last player to add to the total loses

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more
- The last player to add to the total loses

Let's play!

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more
- The last player to add to the total loses

Some states are good; some are bad

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more
- The last player to add to the total loses

Some states are good; some are bad

21

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more
- The last player to add to the total loses

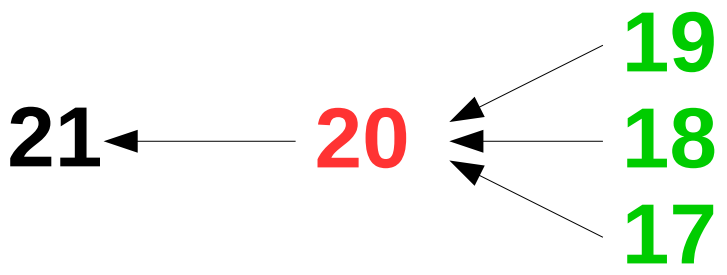
Some states are good; some are bad

21 ◄—— **20**

Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more
- The last player to add to the total loses

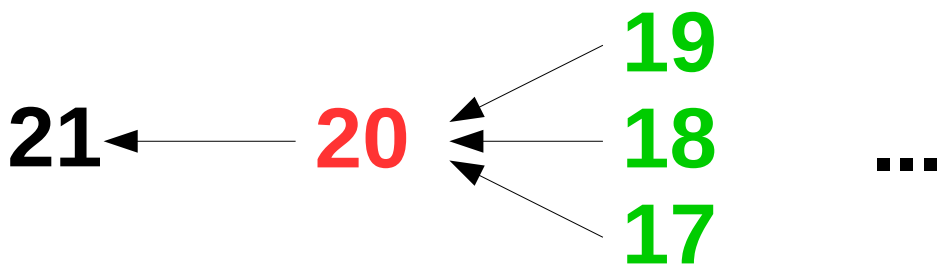
Some states are good; some are bad



Twenty-One Rules

- Two players alternate turns, adding 1, 2, or 3 to the total
- The total starts at 0
- The game ends whenever the total is 21 or more
- The last player to add to the total loses

Some states are good; some are bad



hog

a game of throws

recap

- sequential
- stochastic
- zero-sum
- perfect information

state

- how do we keep track of where we are?

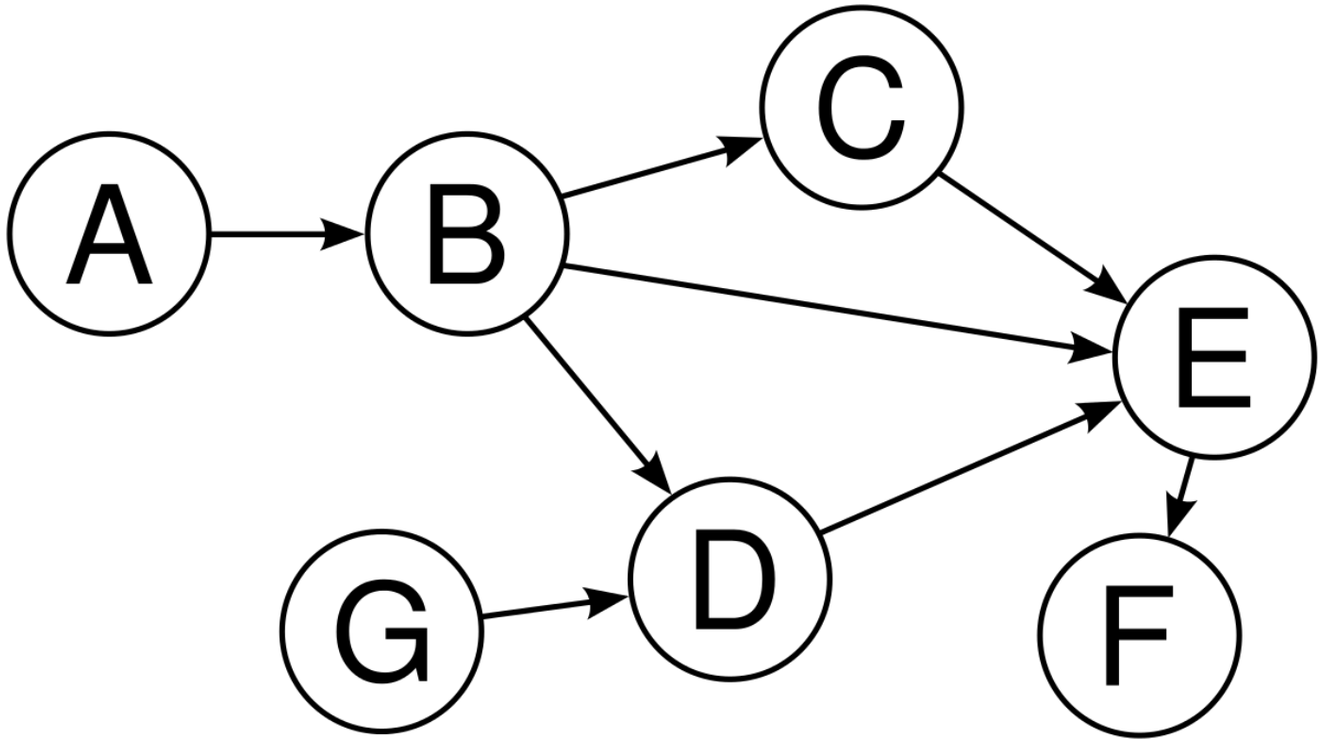
```
(p0_score, p1_score, who)
```

hog strats

- (demo)

graphs

- vertices and edges
- can be directed!



By David W. at de.wikipedia (Original text : David W.) [Public domain], from Wikimedia

hog…is just a graph!

- each state in the game can be expressed as a vertex in the graph

- edges represent transitions between states (whenever a player makes a move)

game trees

- sum of scores is always strictly increasing
- directed acyclic graph (DAG)
  - no cycles
  - sources lead to sinks

probability 101

- expected value

$$\mathbb{E}[X] = \sum_{x} xP(X = x)$$

$$\mathbb{E}[S_n] = \sum_{s=0}^{6n} sP(S_n = s)$$

calculating winrates

- run thousands of experiments (empirical)
- use the power of expectation! (analytic)

calculating winrates

- Let $P(s0, s1)$ be the probability of winning when it's player 0's turn, and the scores are s0 to s1

$$P(s0, s1) = \sum_{s} P(S_n = s)(1 - P(s1, s0 + s))$$

$$P(s0 \geq 100, s1) = 1 \qquad P(s0, s1 \geq 100) = 0$$

probabilistic algorithms

- Nim (minimax)
  - *minimize* the loss from the worst-case scenario
- Hog (expectiminimax)
  - now there's randomness!
  - minimize the *expectation* of the worst-case scenario

countering strategies

- (demo)

contest rules

- *Pork Chop*: You get one chance to force a switch with your opponent

```
(p0_score, p1_score, who,
```

church numerals

electric boogaloo

church numerals

- what do numbers actually represent?

$$f^n(x)$$

successor

$$f^{n+1}(x) = f(f^n(x))$$

add

$$f^{m+n}(x) = f^m(f^n(x))$$

multiply

$$f^{mn}(x) = (f^m)^n(x)$$

exponentiation

$$n(m)(f) = m^n(f)$$

other operations?

- *predecessor* operation is opposite of *successor*
- subtraction, division $pred(f^n(x)) = f^{n-1}x$
- signed numbers!
- boolean logic!
- rationals, reals, and complex numbers???!!!11

that's all, folks

hog contest > midterm1