61A Extra Lecture 2

Thursday, February 5

Announcements

Announcements

- If you want 1 unit (pass/no pass) of credit for this CS 98, you need to:

Announcements

- If you want 1 unit (pass/no pass) of credit for this CS 98, you need to:
  - Enroll in "Additional Topics on the Structure and Interpretation of Computer Programs"

Announcements

- If you want 1 unit (pass/no pass) of credit for this CS 98, you need to:
  - Enroll in "Additional Topics on the Structure and Interpretation of Computer Programs"
  - Course control number: 25709

Announcements

- If you want 1 unit (pass/no pass) of credit for this CS 98, you need to:
  - Enroll in "Additional Topics on the Structure and Interpretation of Computer Programs"
  - Course control number: 25709
- Extra Homework 1 due Thursday 2/12 @ 11:59pm

Dice

Hog: The End Game

Hog: The End Game

You: 98
Them: 99

Hog: The End Game

You: 98
Them: 99

You: 92
Them: 99

Hog: The End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

Hog: The End Game

You: 98
Them: 99

You: 92
Them: 99

You: 88
Them: 99

You: 80
Them: 99

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

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

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

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

(assuming k > 1)

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

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

(assuming k > 1)

The chance to score at least **k** in **n** rolls can be computed using tree recursion!

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

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

(assuming k > 1)

The chance to score at least **k** in **n** rolls can be computed using tree recursion!

Sum over each possible dice outcome **t** that does not *pig out*:
The chance to roll **t** times the chance to score at least **k - t** points using **n - 1** rolls.

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

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

(assuming k > 1)

The chance to score at least **k** in **n** rolls can be computed using tree recursion!

Sum over each possible dice outcome **t** that does not *pig out*:
    The chance to roll **t** times the chance to score at least **k - t** points using **n - 1** rolls.

Base case: The chance to score at least 0 in 0 rolls is 1 (guaranteed)

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

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

(assuming k > 1)

The chance to score at least **k** in **n** rolls can be computed using tree recursion!

Sum over each possible dice outcome **t** that does not *pig out*:
    The chance to roll **t** times the chance to score at least **k - t** points using **n - 1** rolls.

Base case: The chance to score at least 0 in 0 rolls is 1 (guaranteed)

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

Hog: The End Game

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

What is the chance that I'll score at least **k** points rolling **n** six-sided dice?

S_n : Score from rolling n dice

t : A single outcome of rolling once

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

(assuming k > 1)

The chance to score at least **k** in **n** rolls can be computed using tree recursion!

Sum over each possible dice outcome **t** that does not *pig out*:
    The chance to roll **t** times the chance to score at least **k - t** points using **n - 1** rolls.

Base case: The chance to score at least 0 in 0 rolls is 1 (guaranteed)

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

(Demo)

Memoization

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

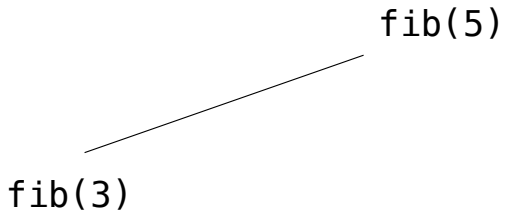
fib(5)

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

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

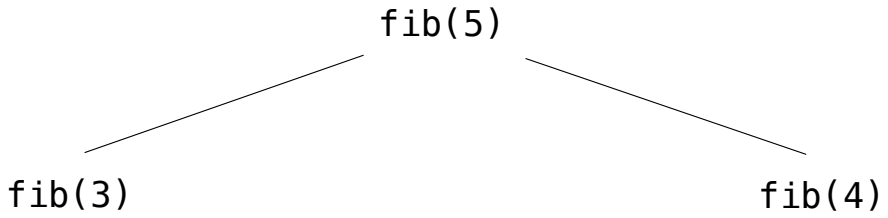



```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

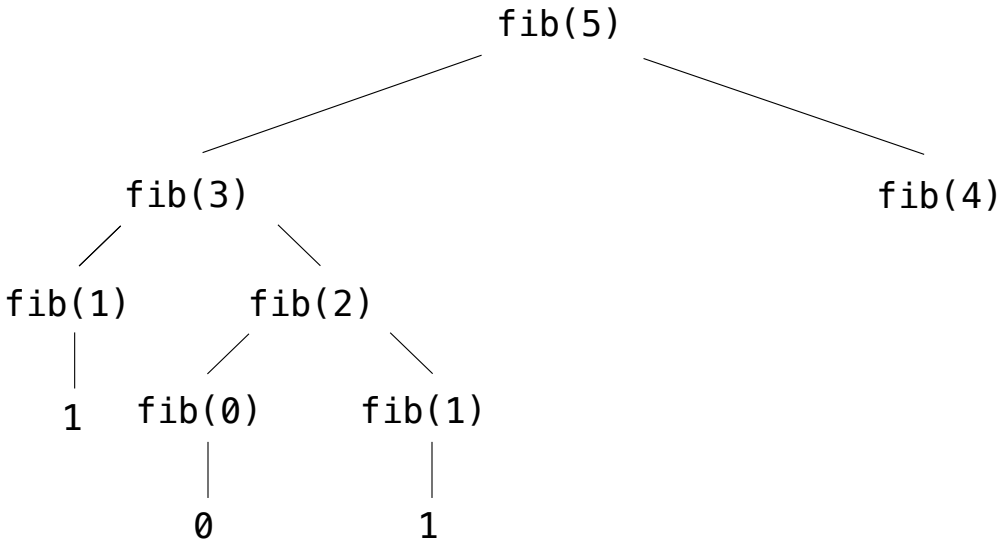



```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:



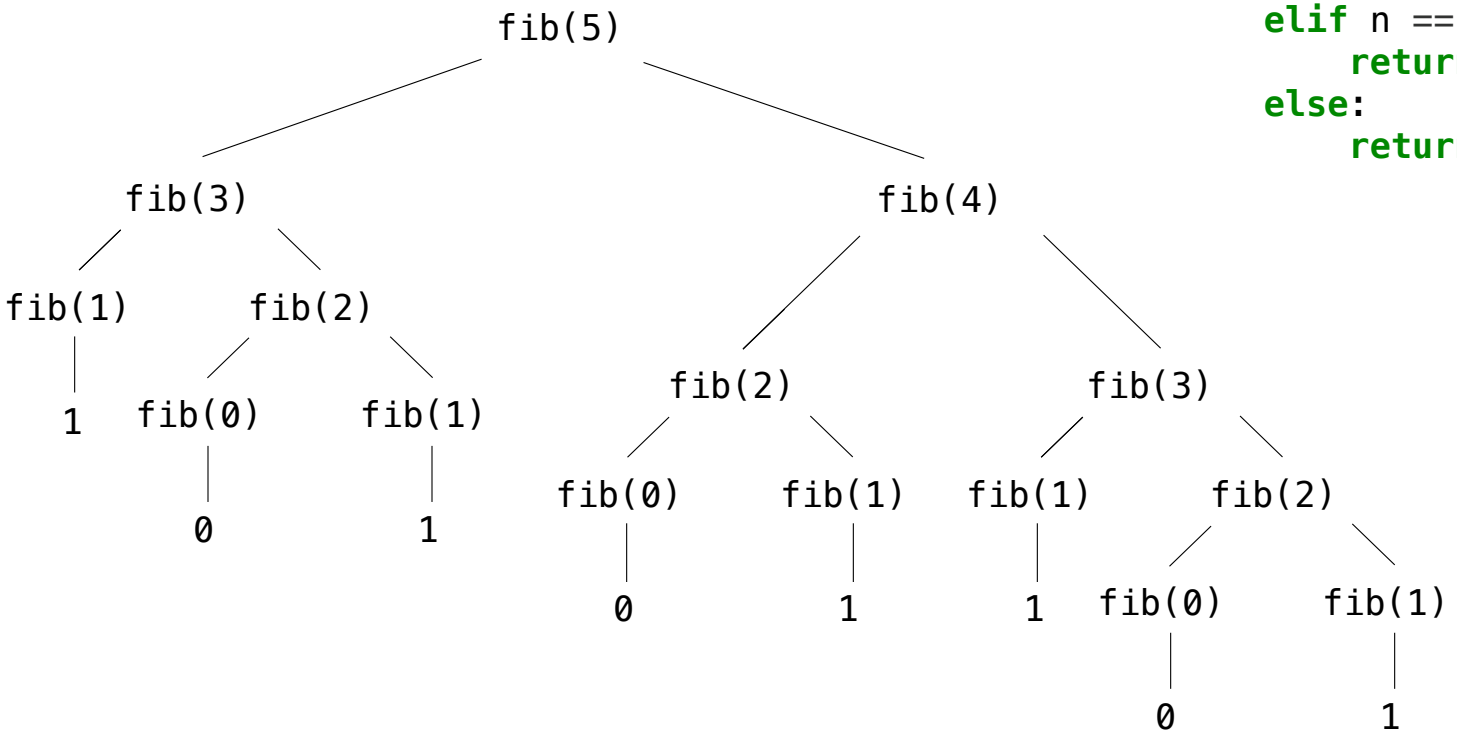

```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

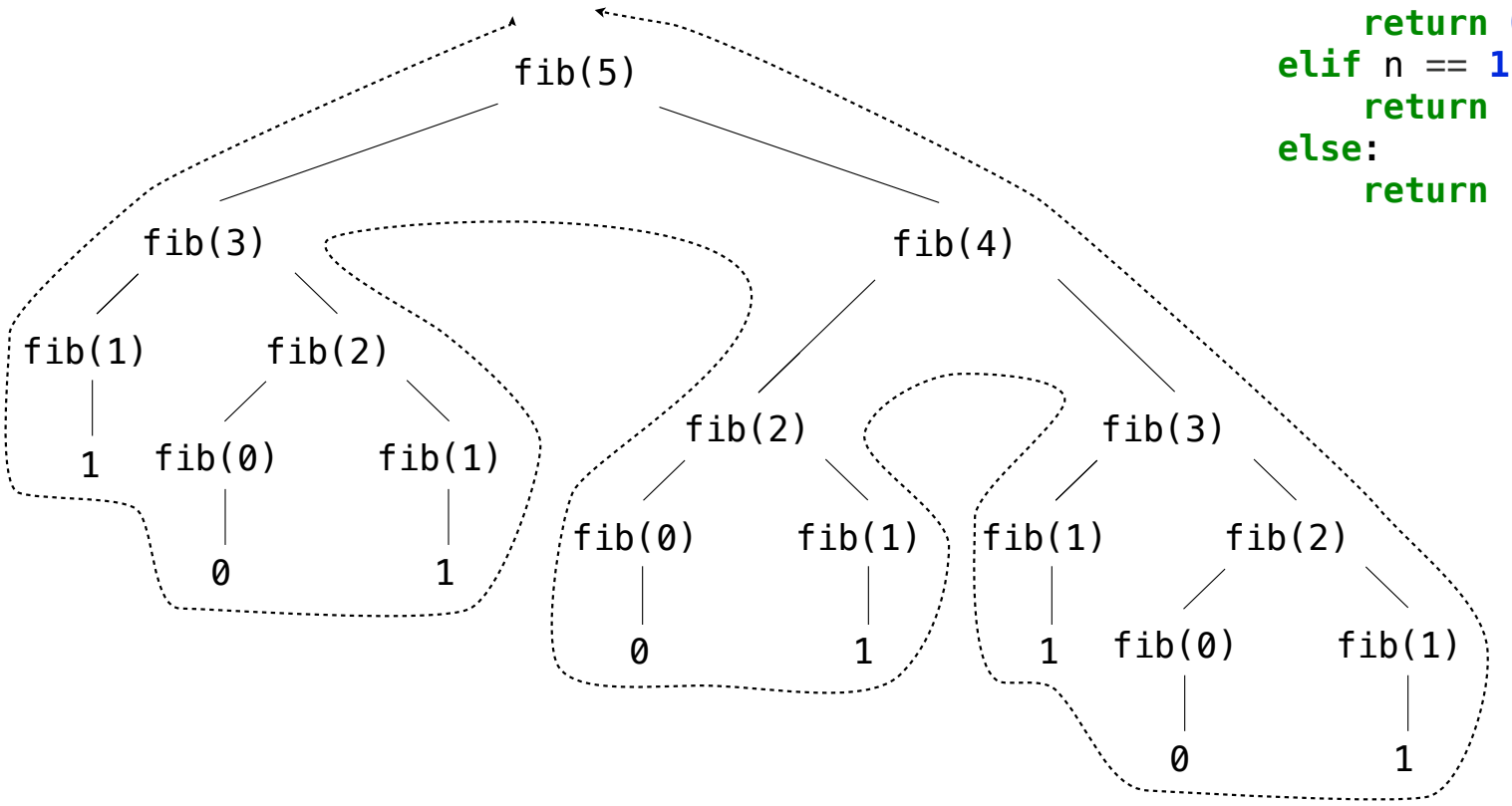
```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```



http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:



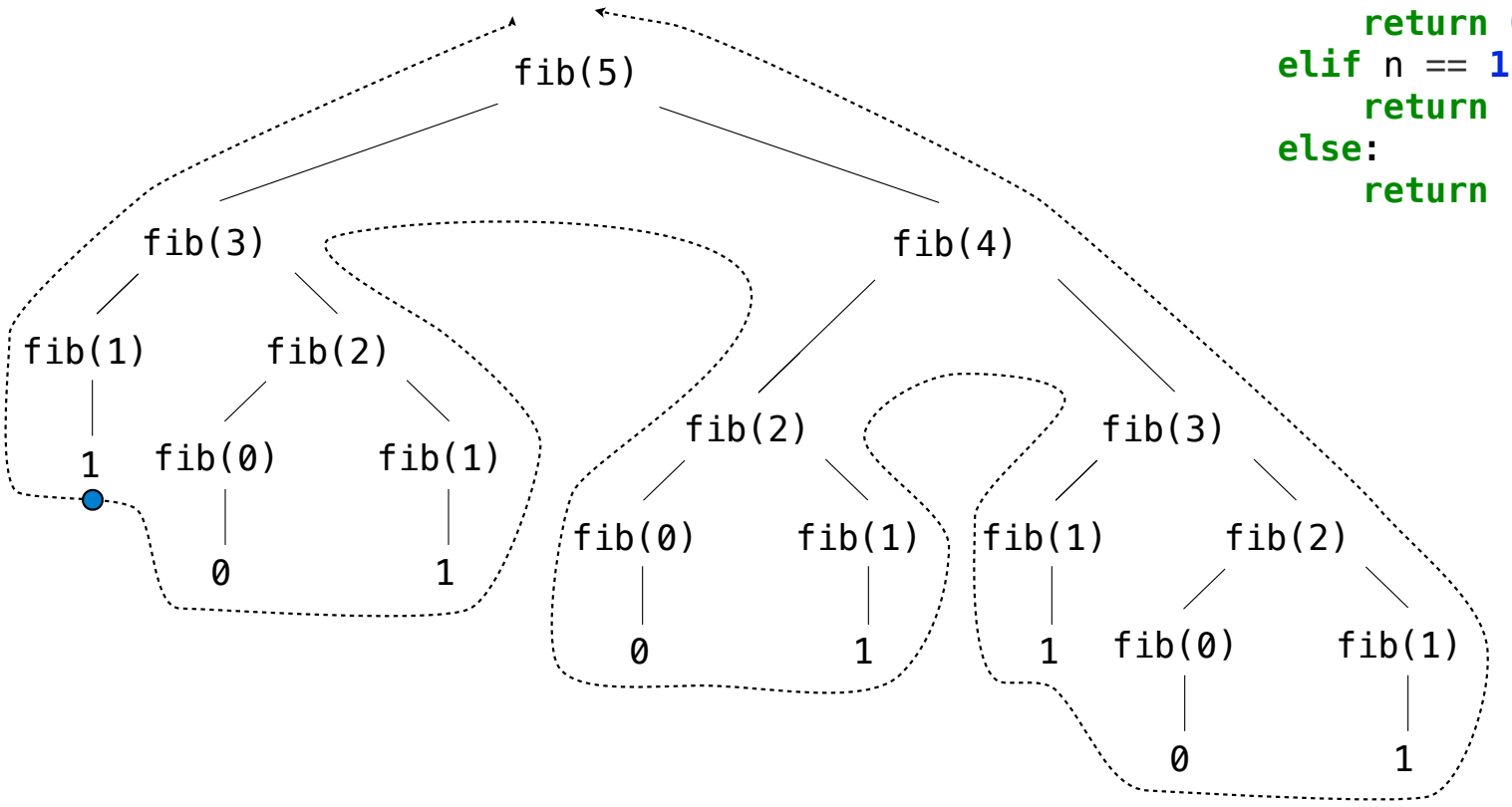

```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

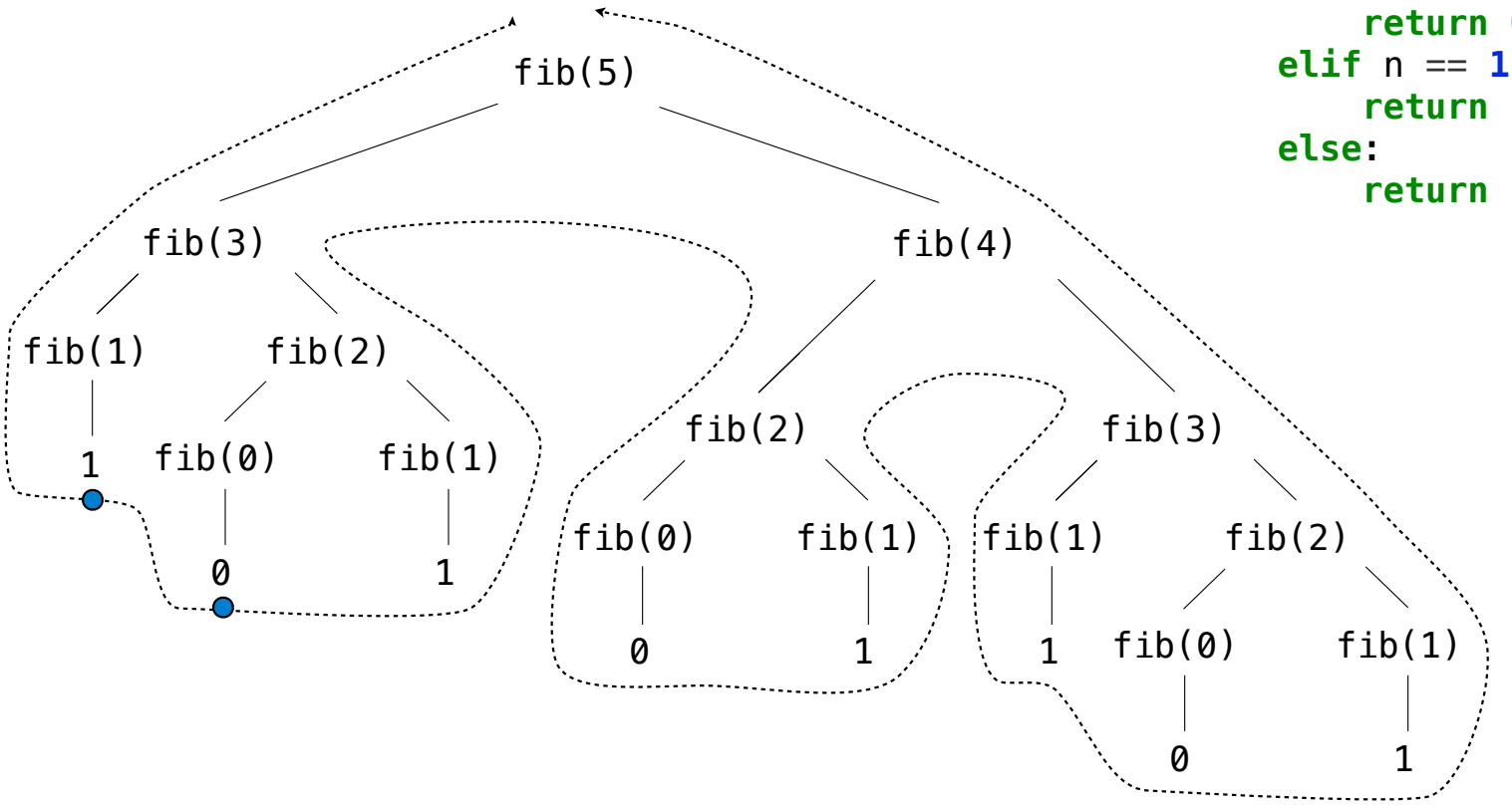
```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```



http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:



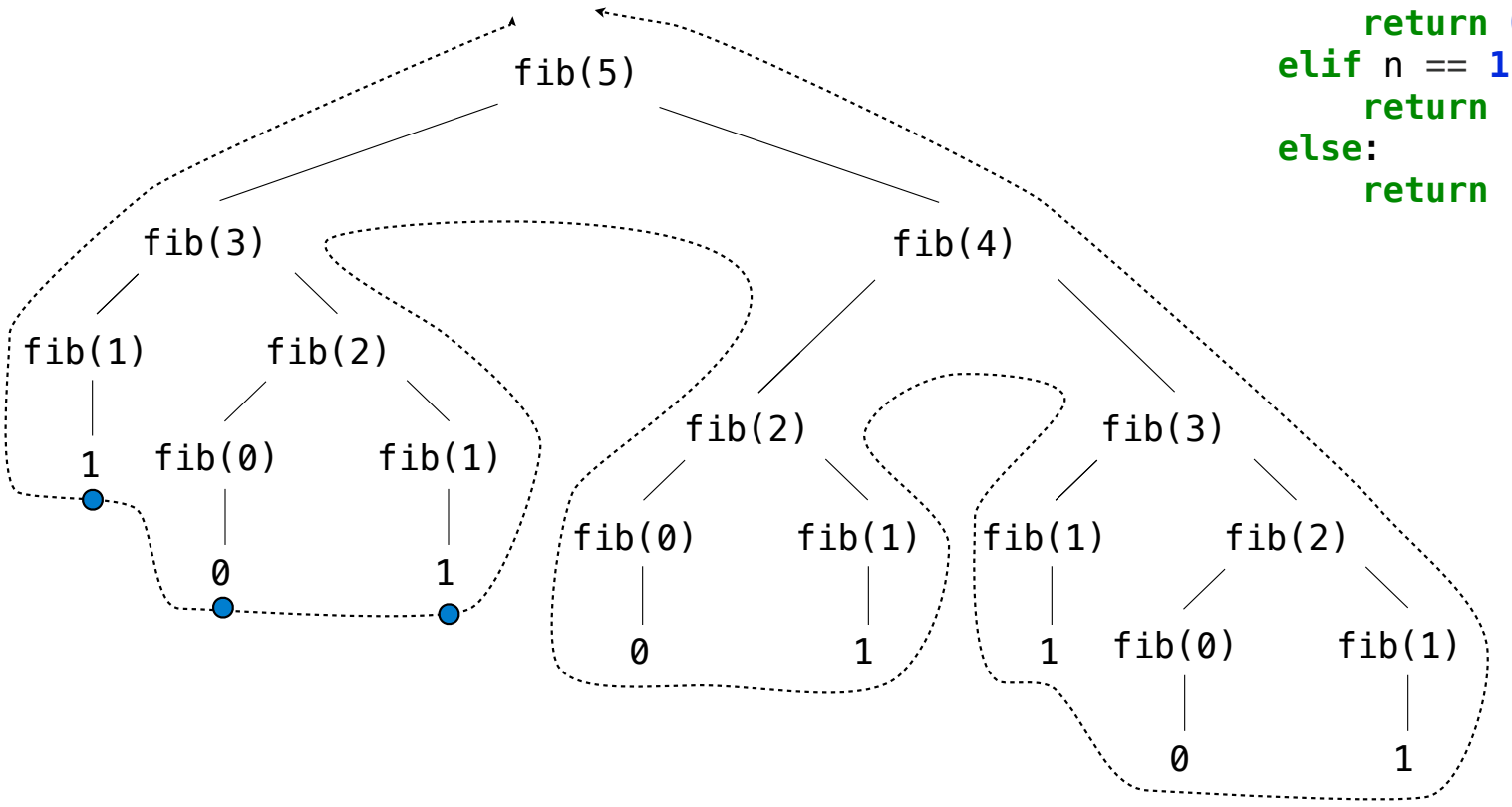

```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

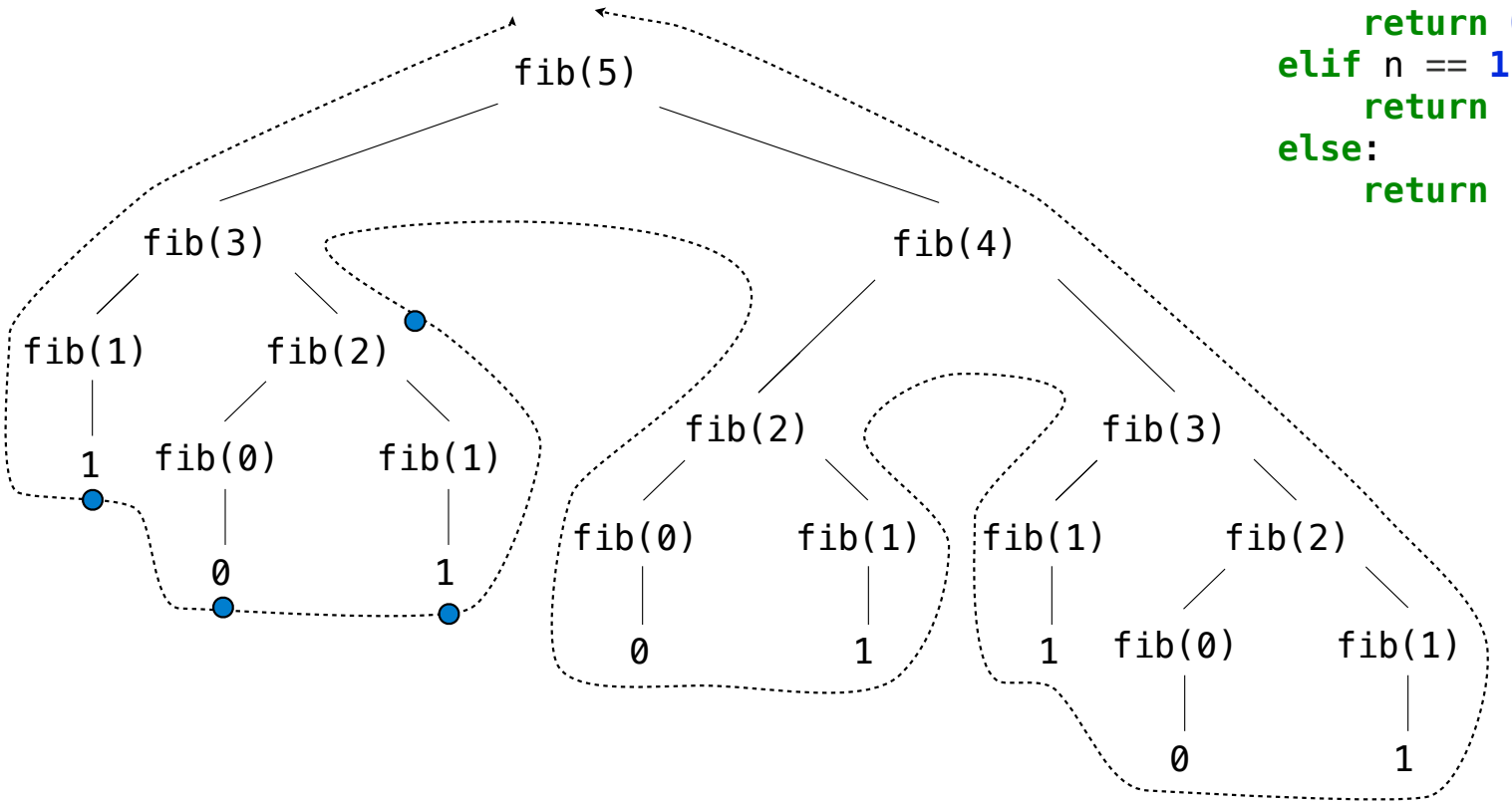


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

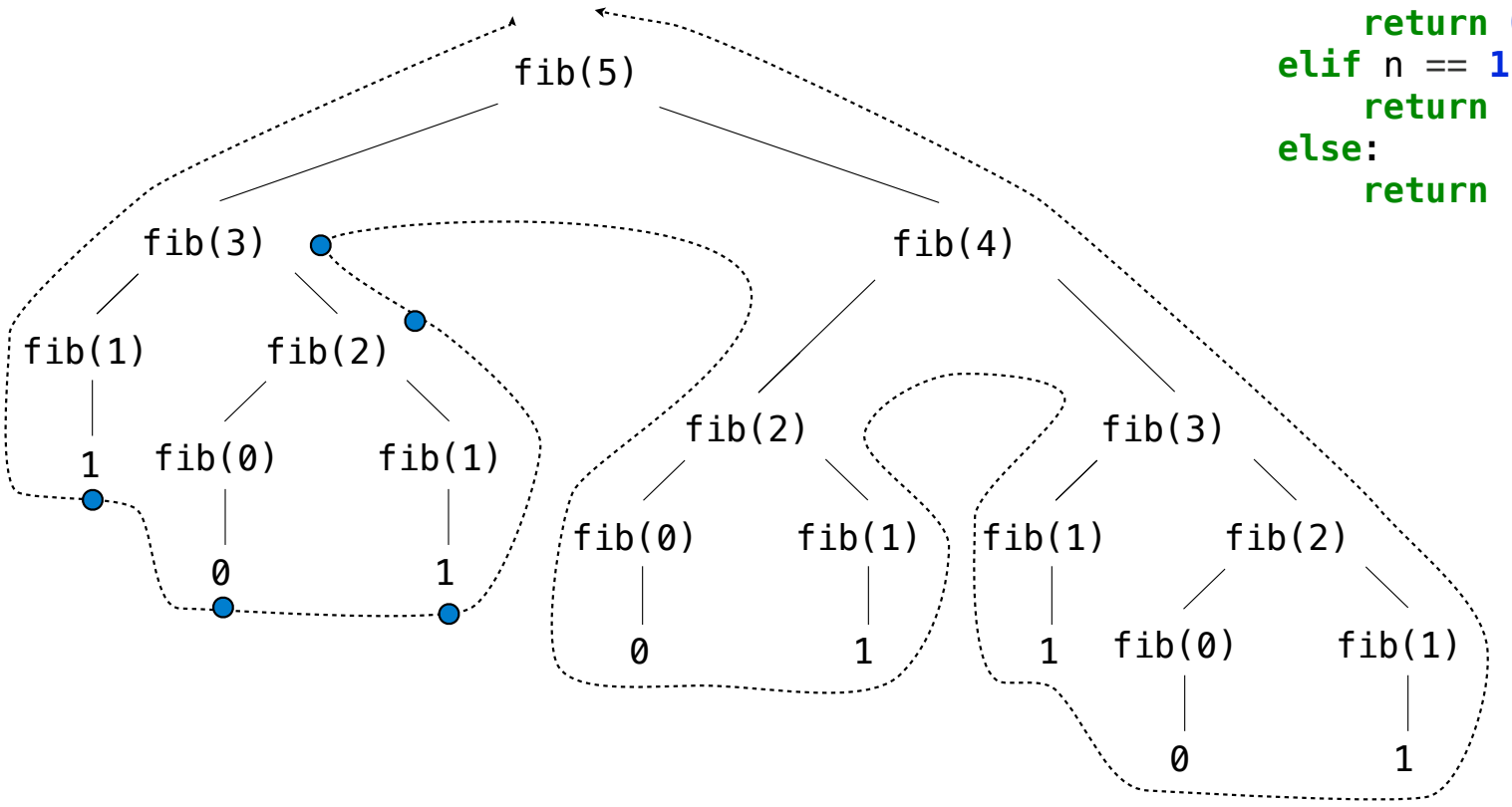


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

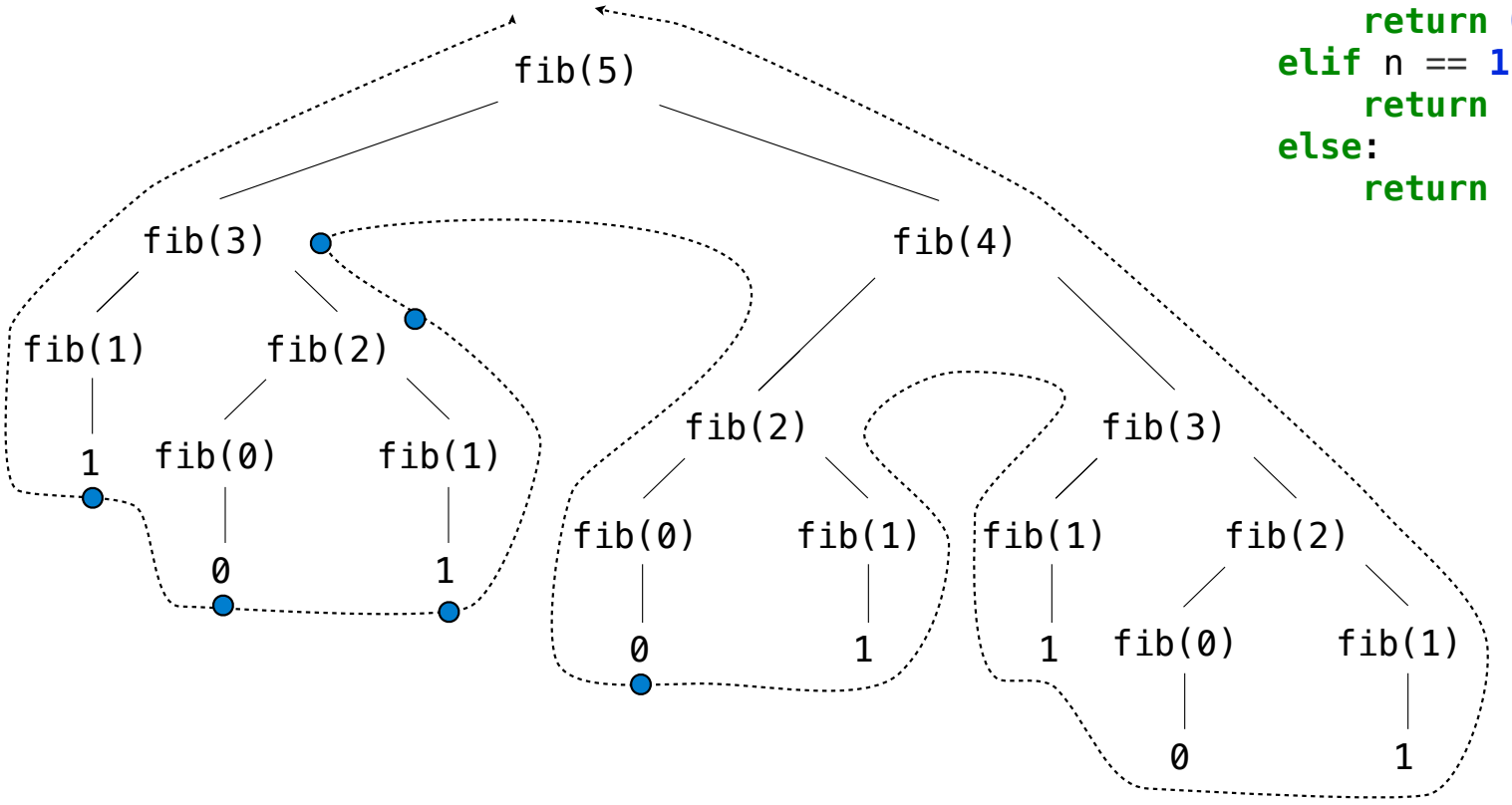


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

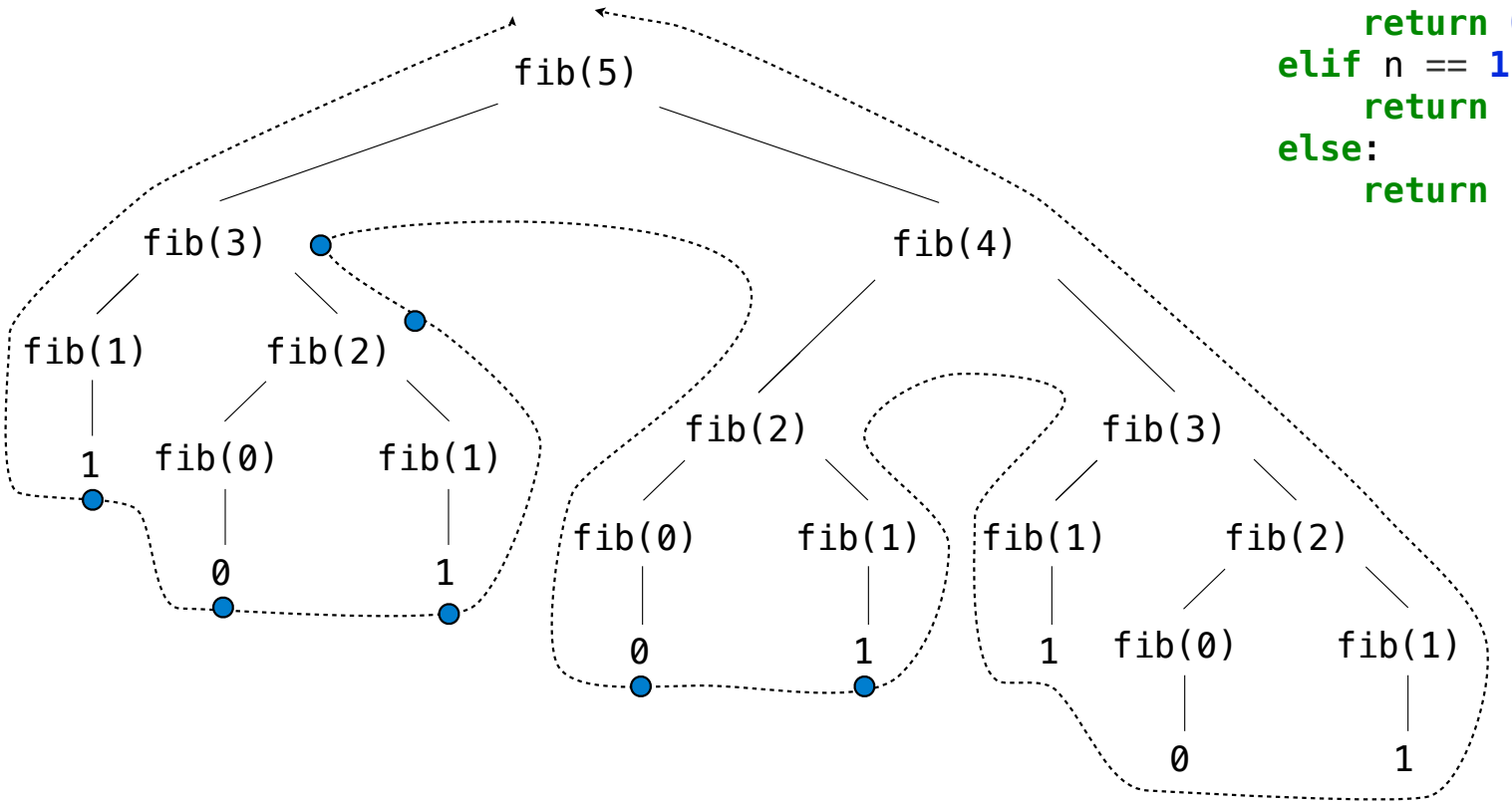


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

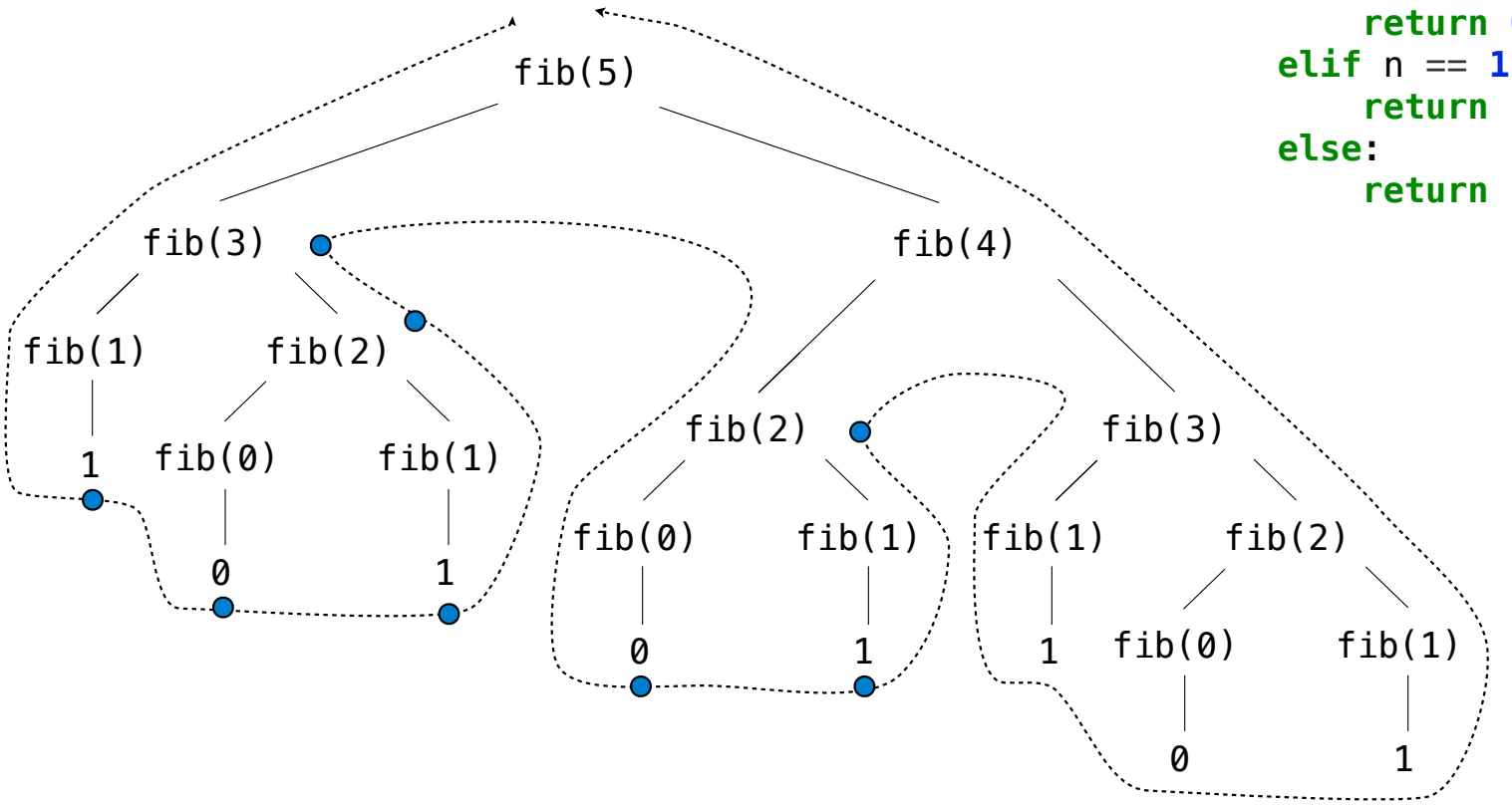


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

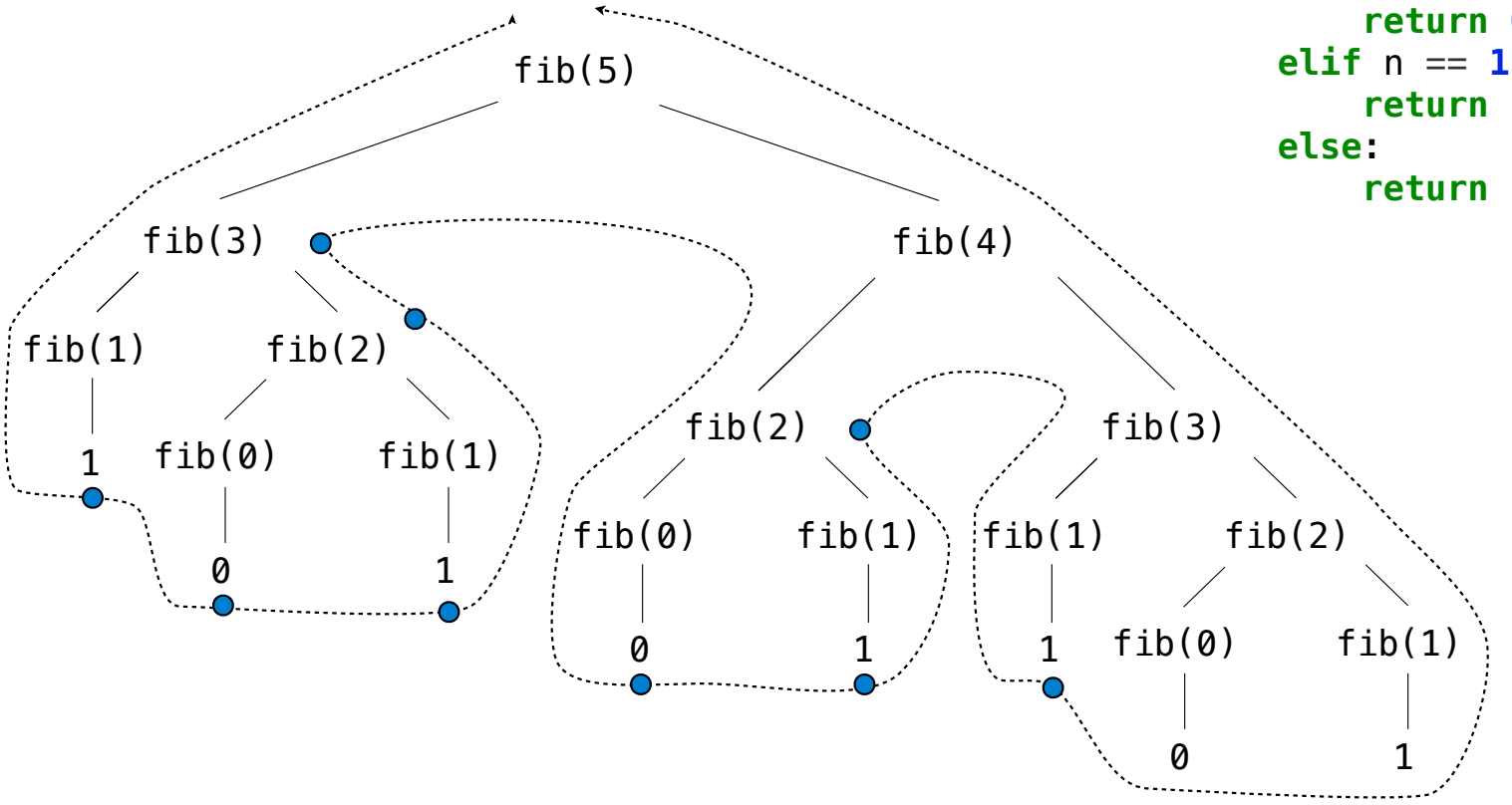


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

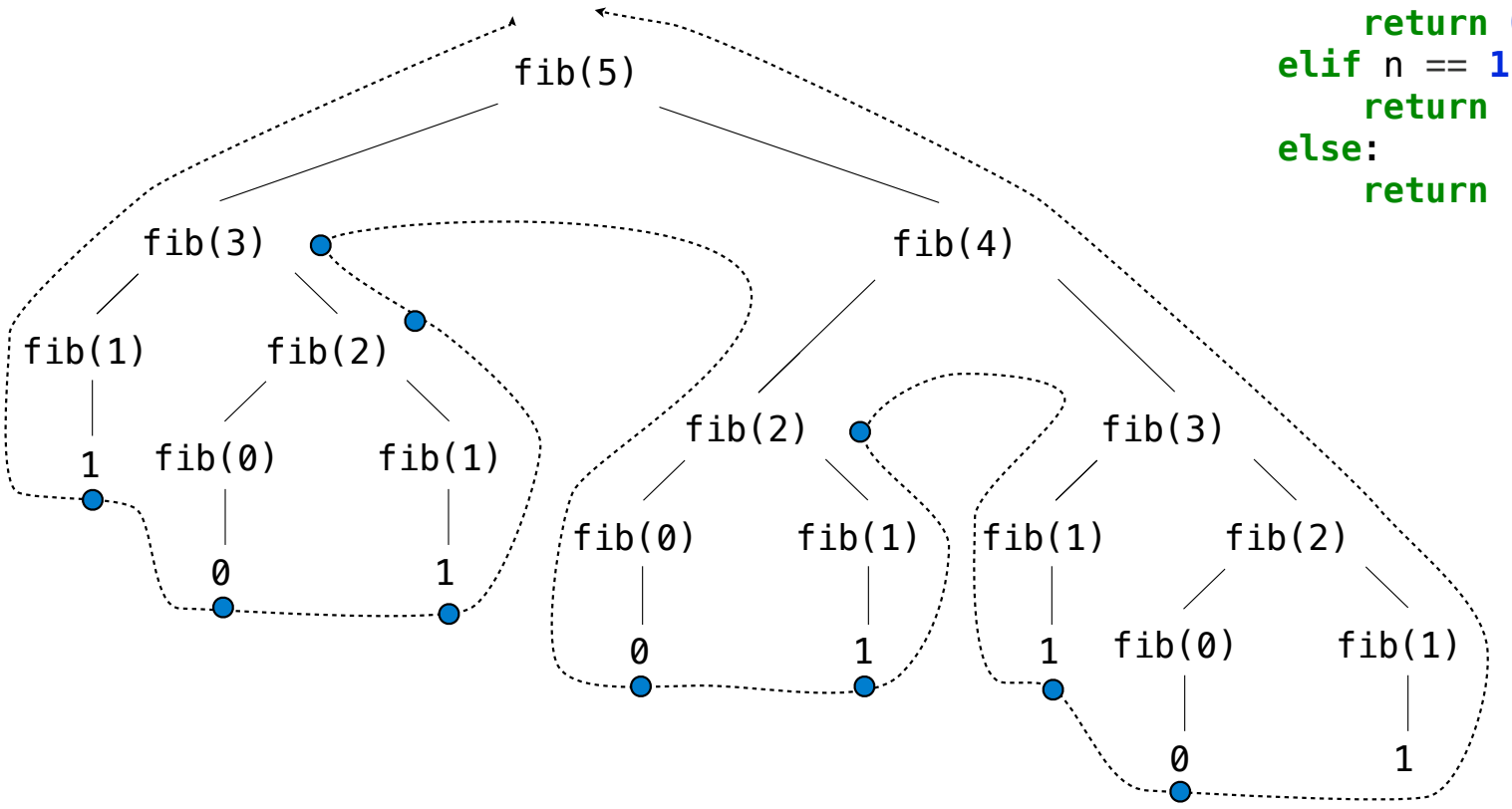


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

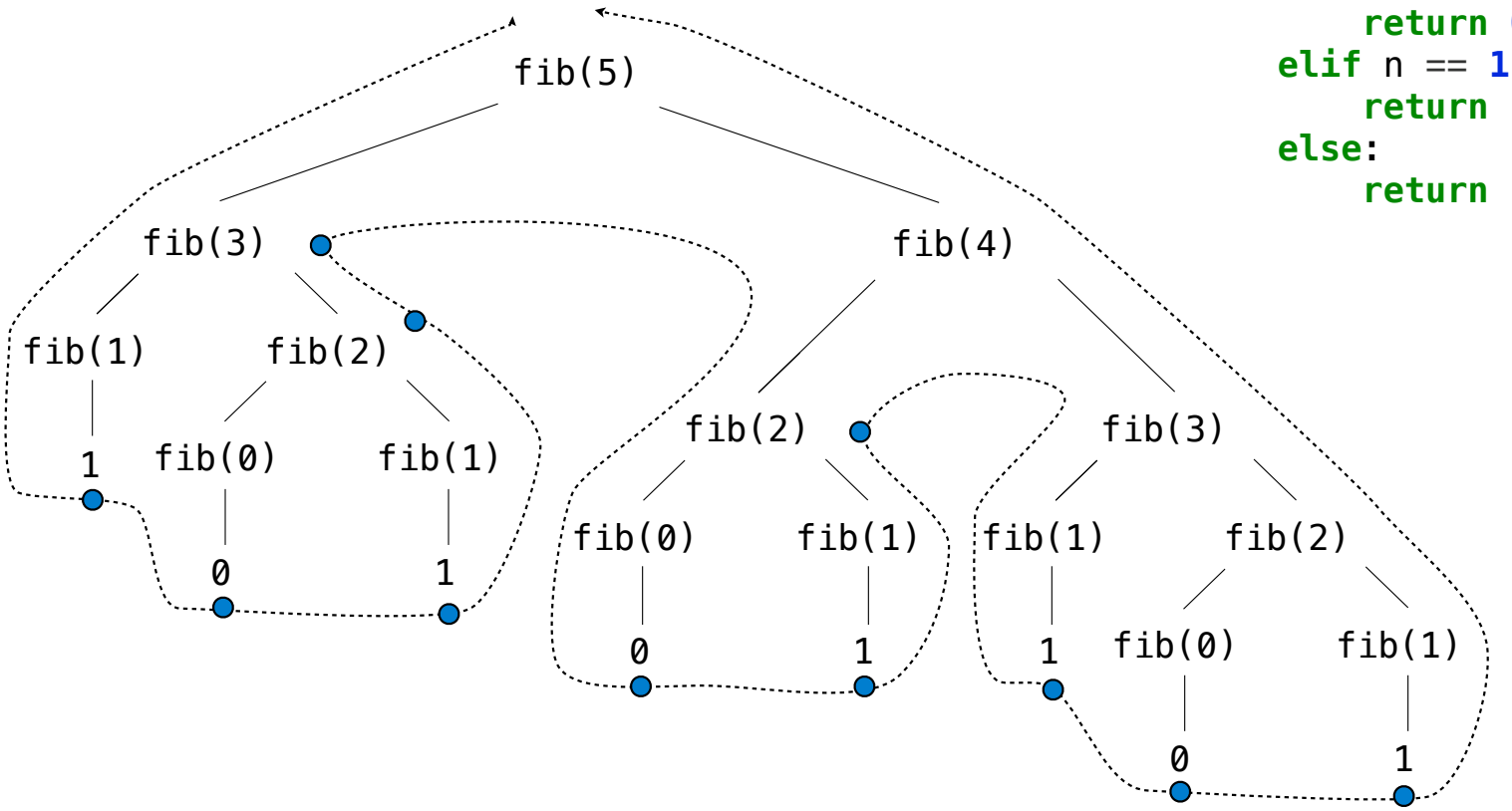


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

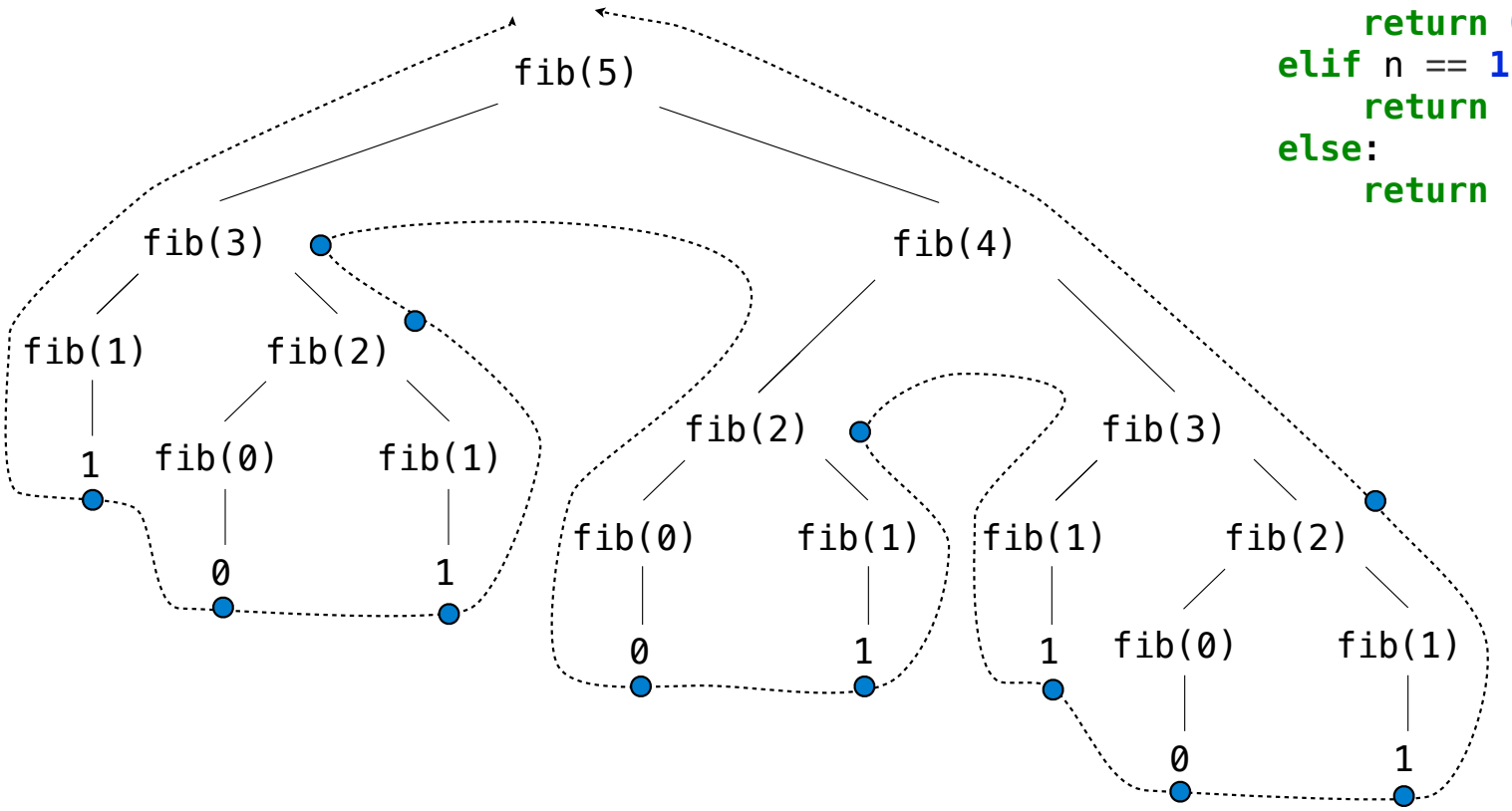


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

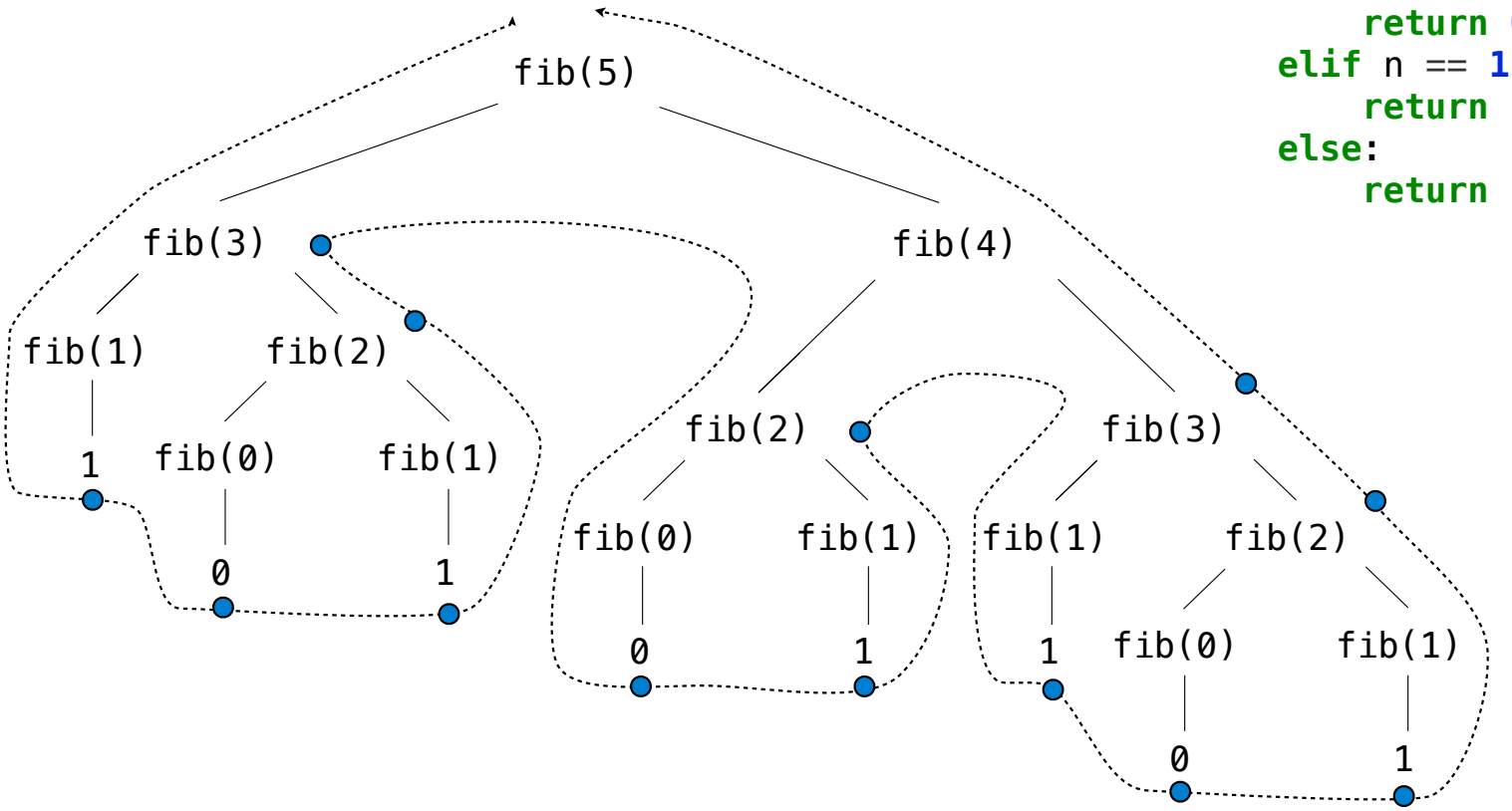


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

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

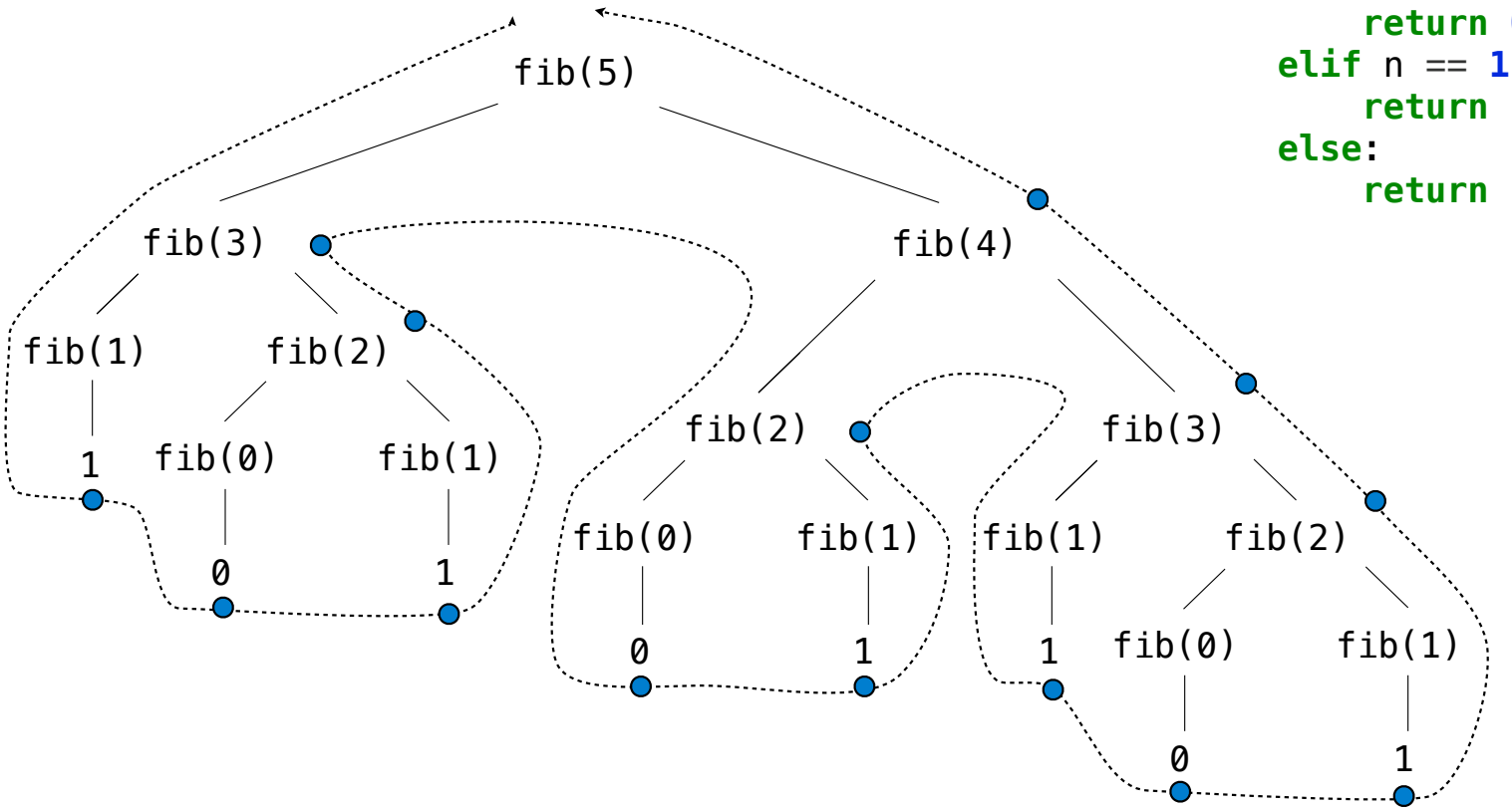


http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

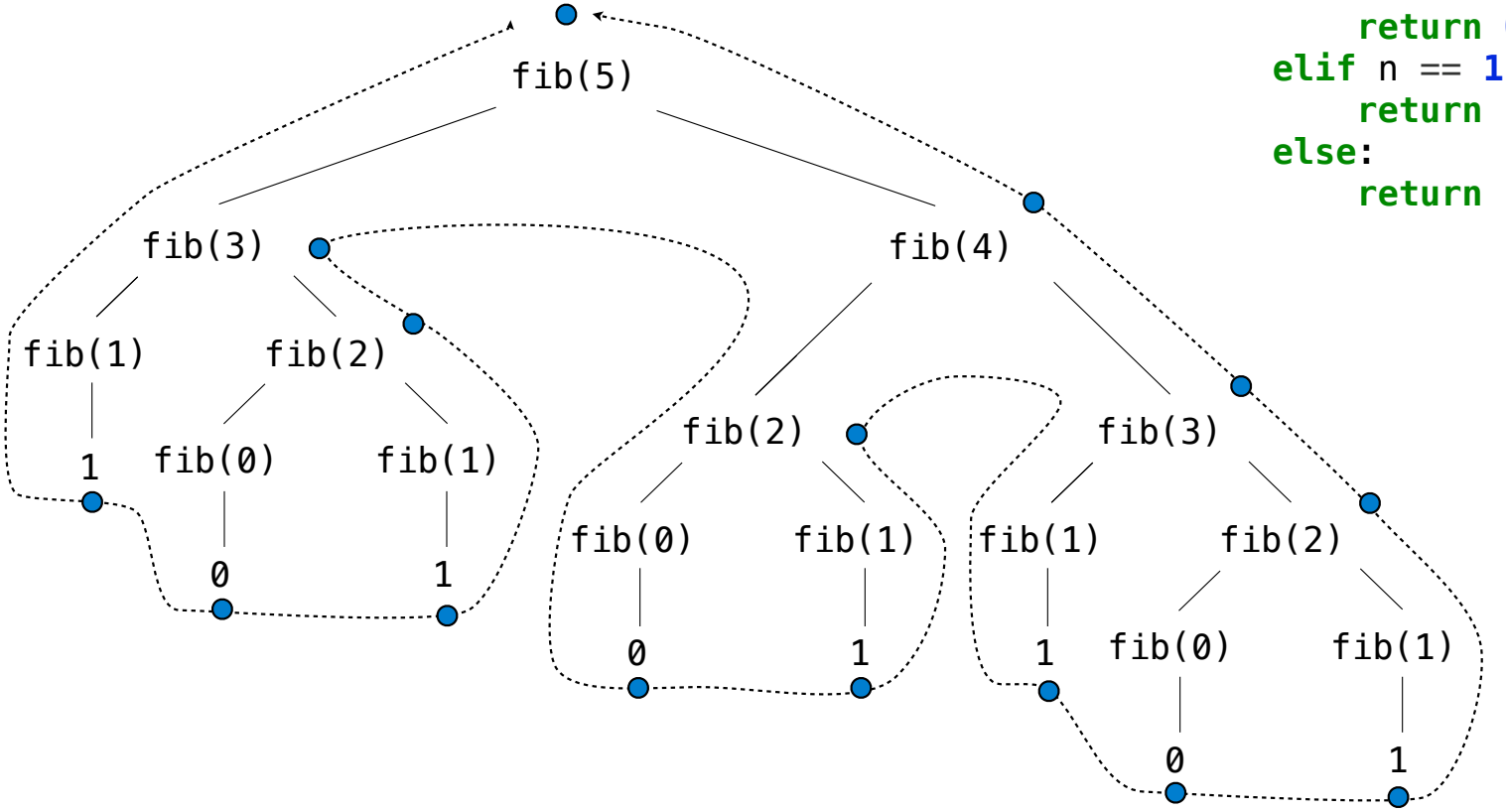
```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```



http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

Our first example of tree recursion:

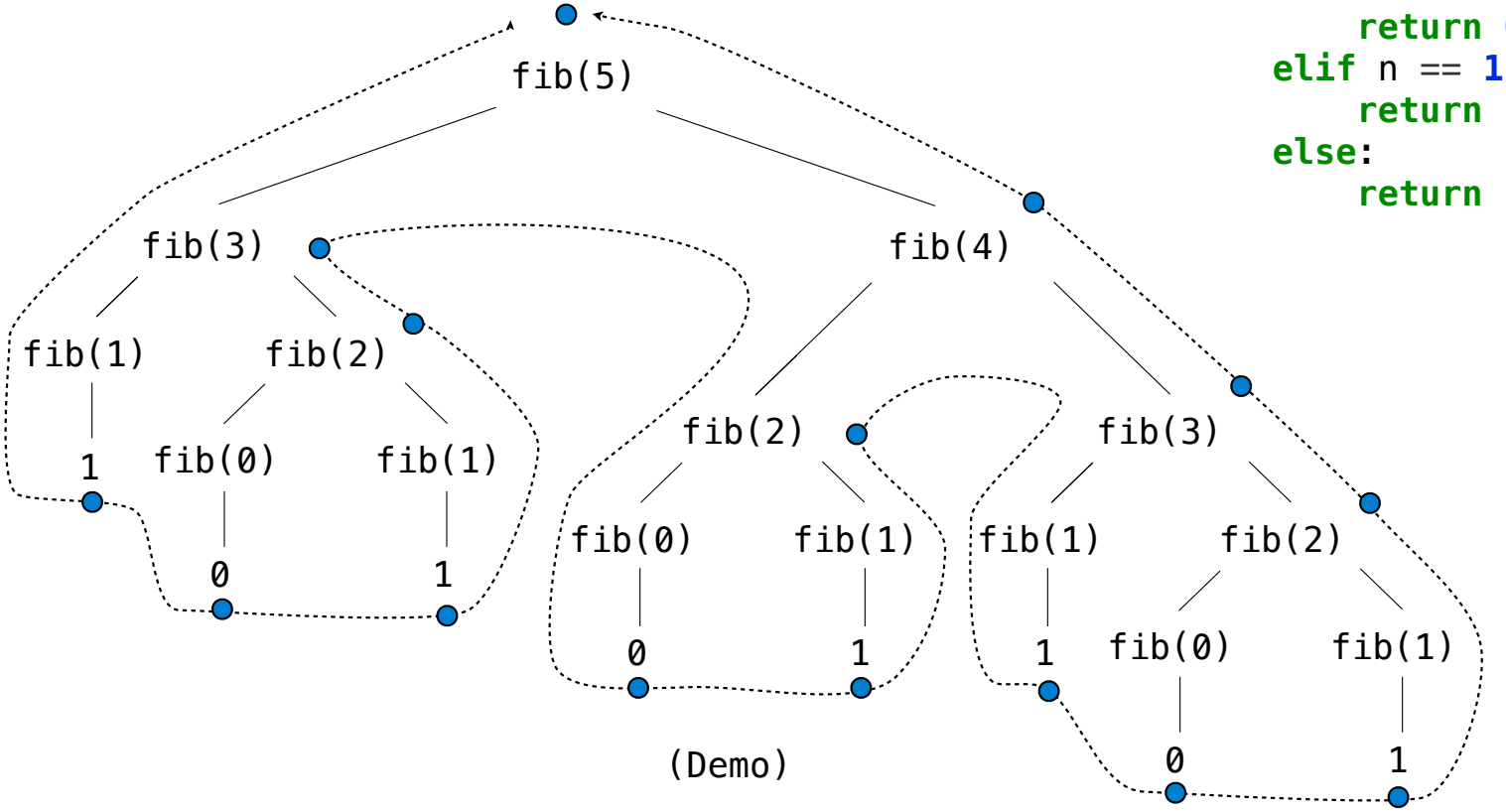



```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-2) + fib(n-1)
```

http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Recursive Computation of the Fibonacci Sequence

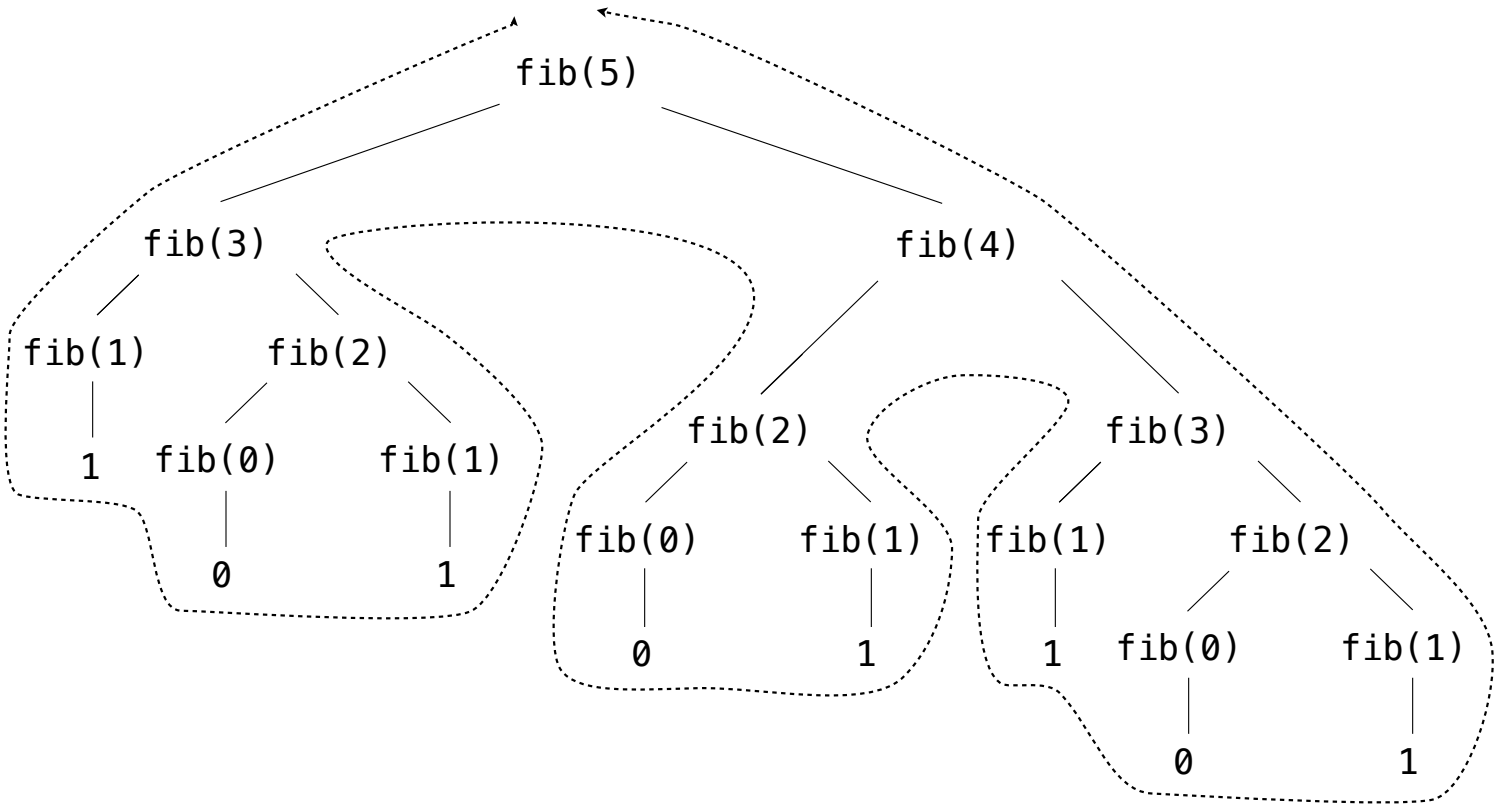
Our first example of tree recursion:




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

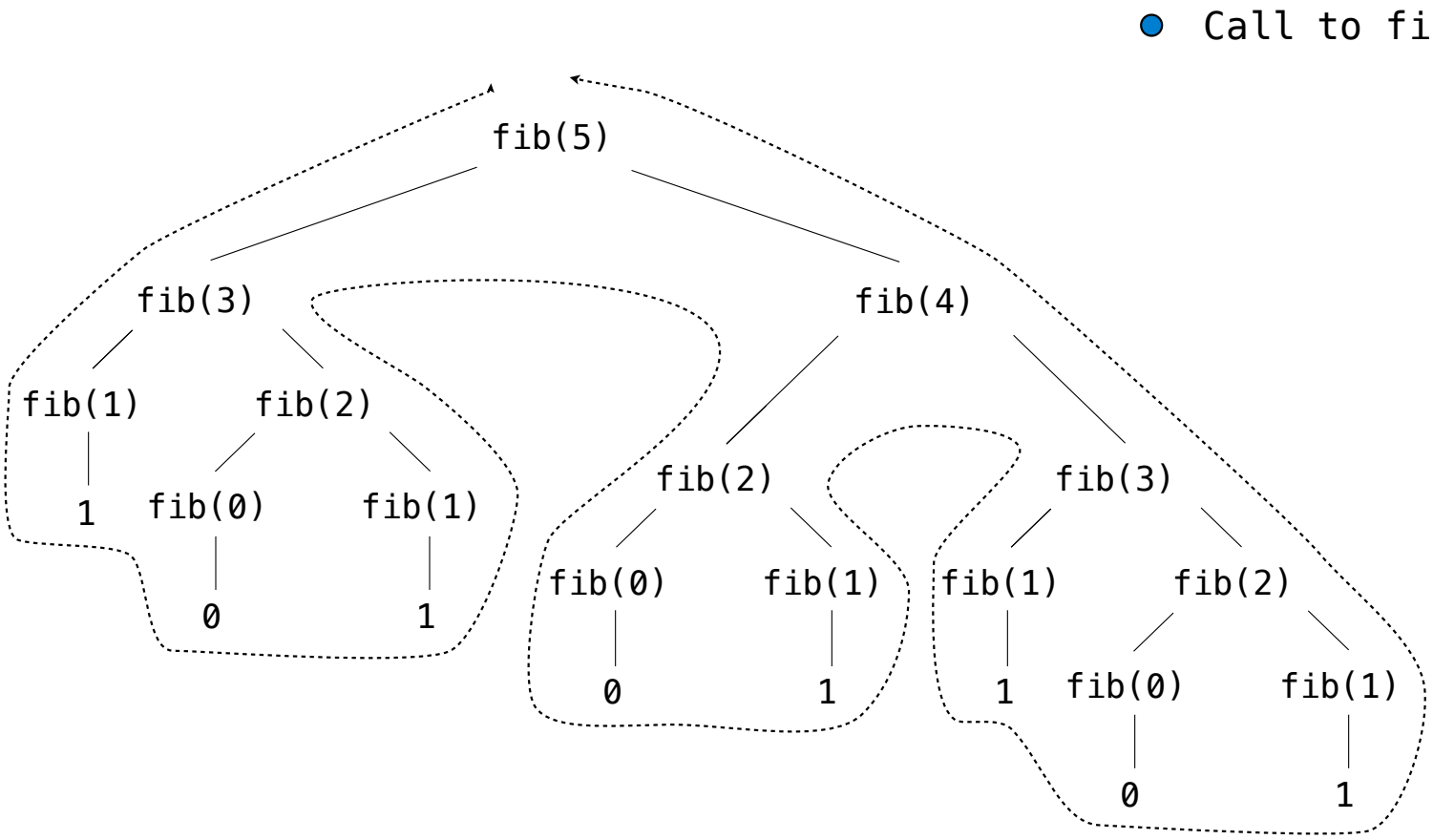
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



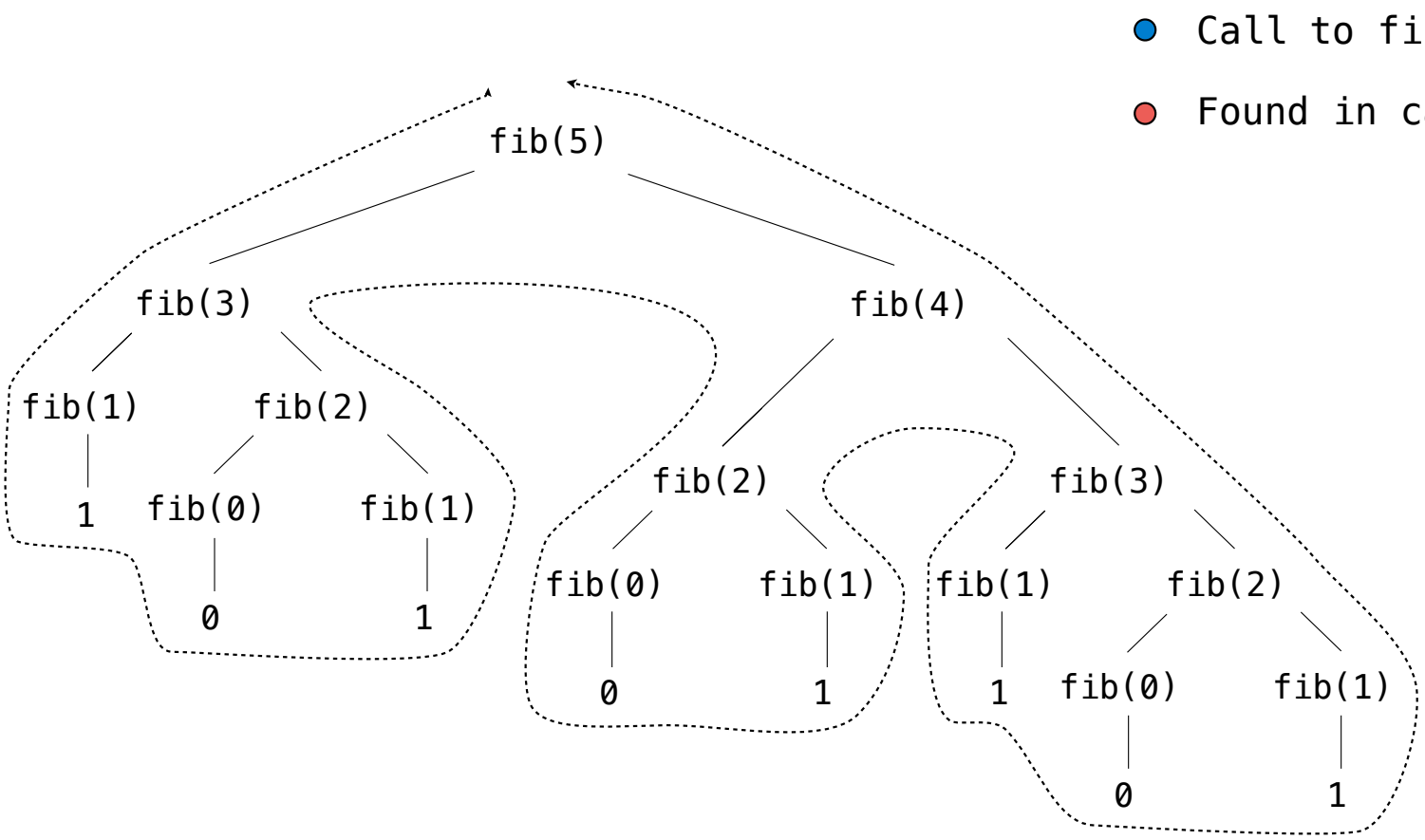
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



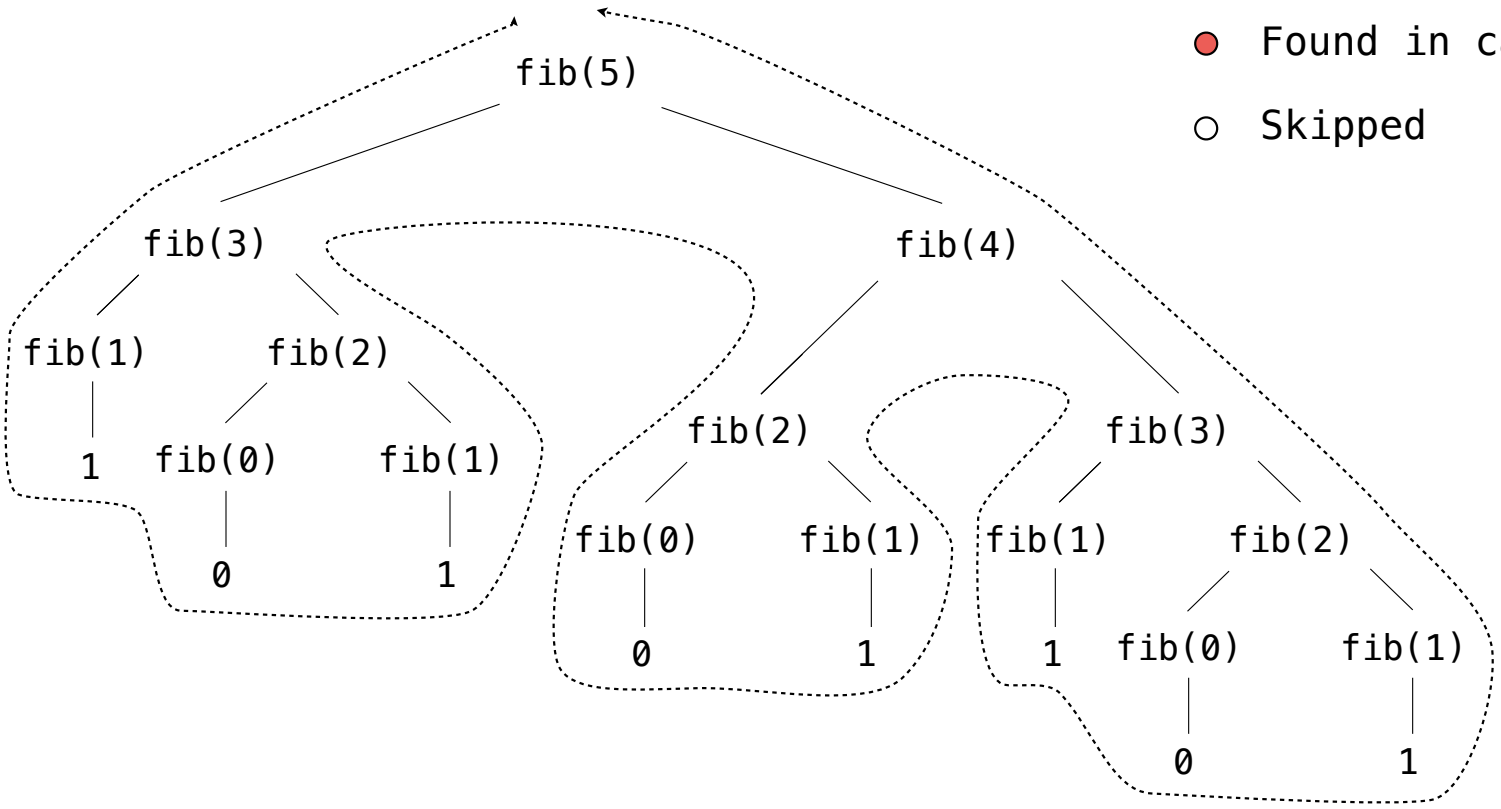
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



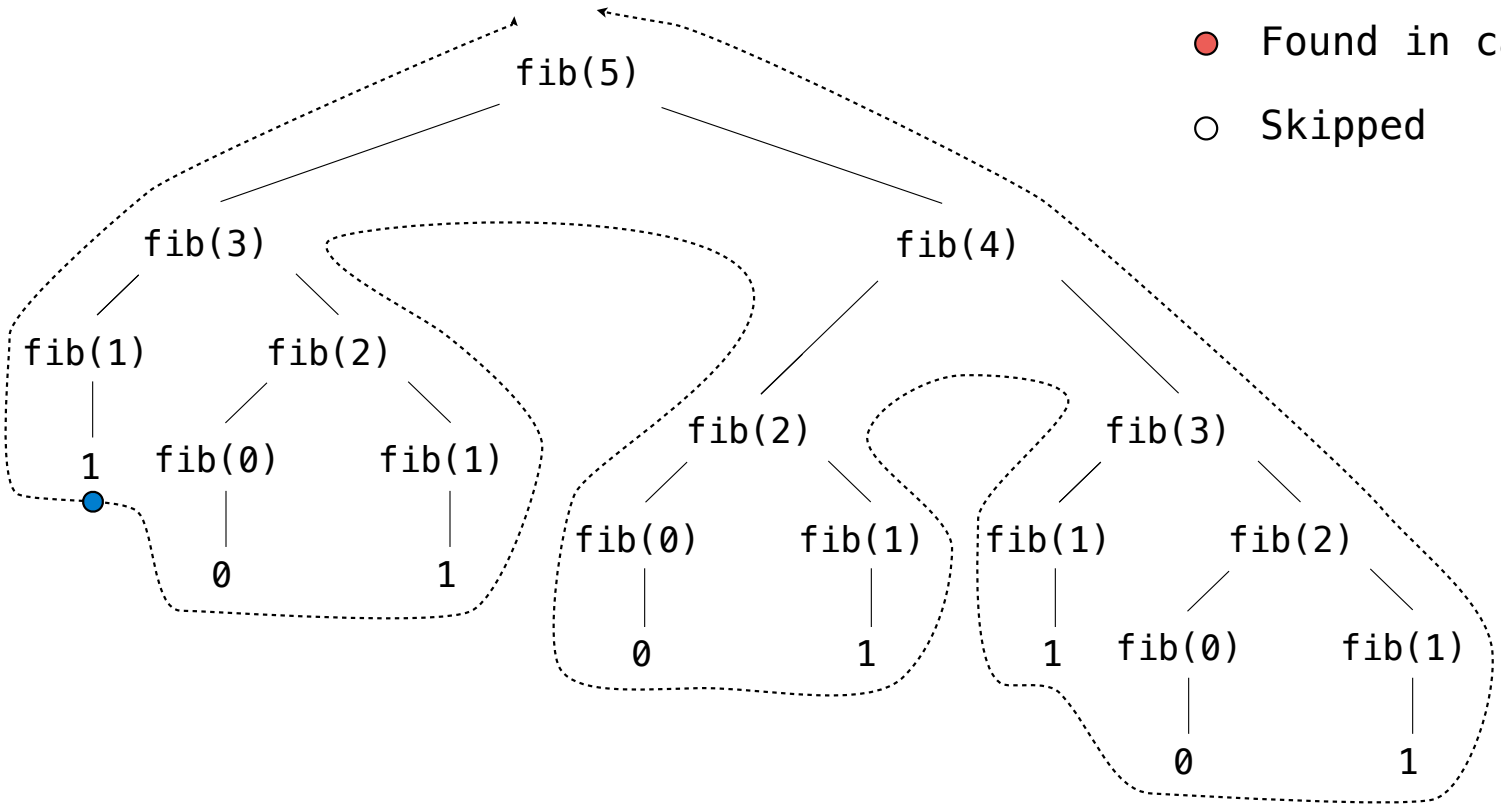
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



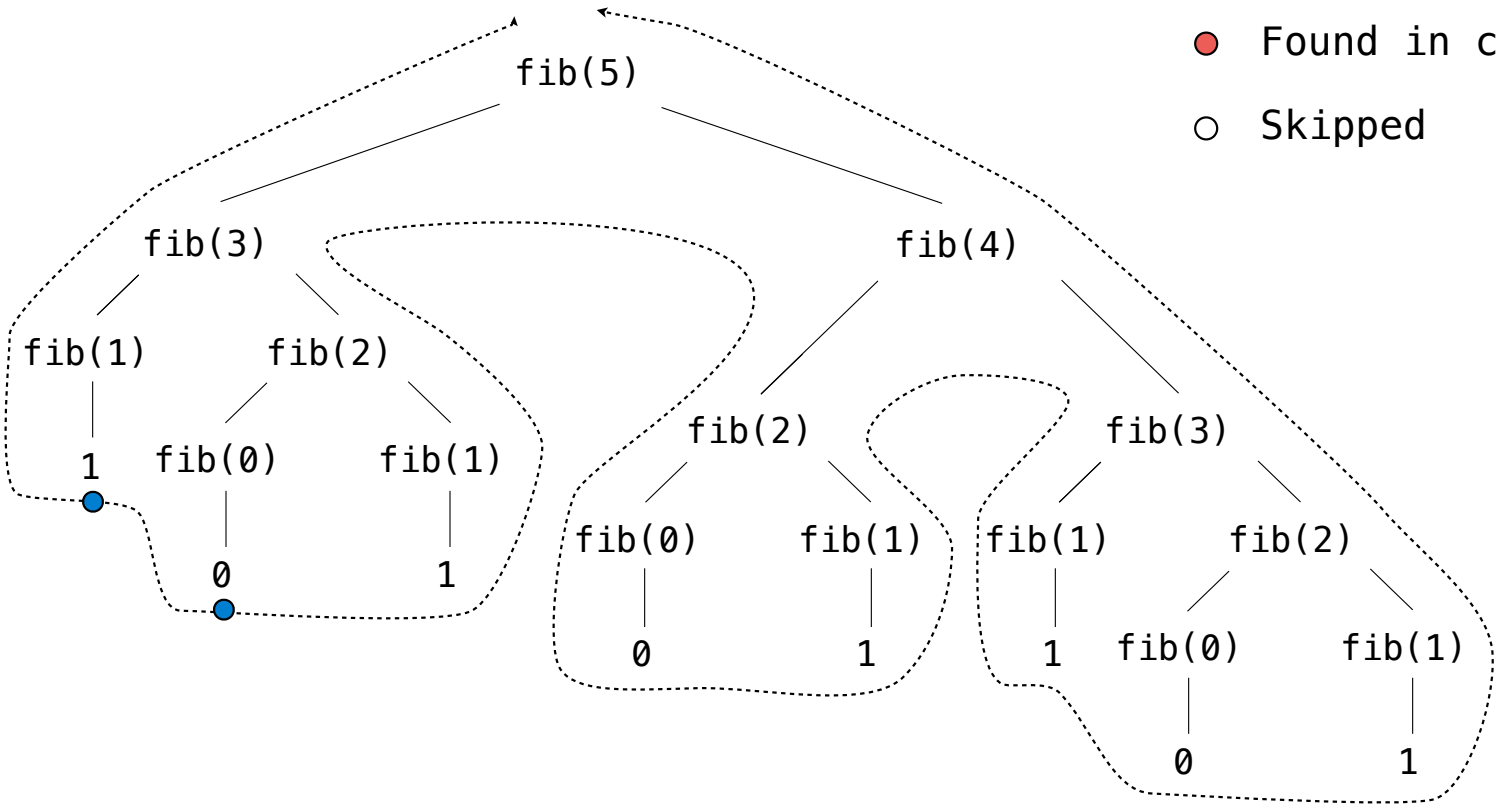
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



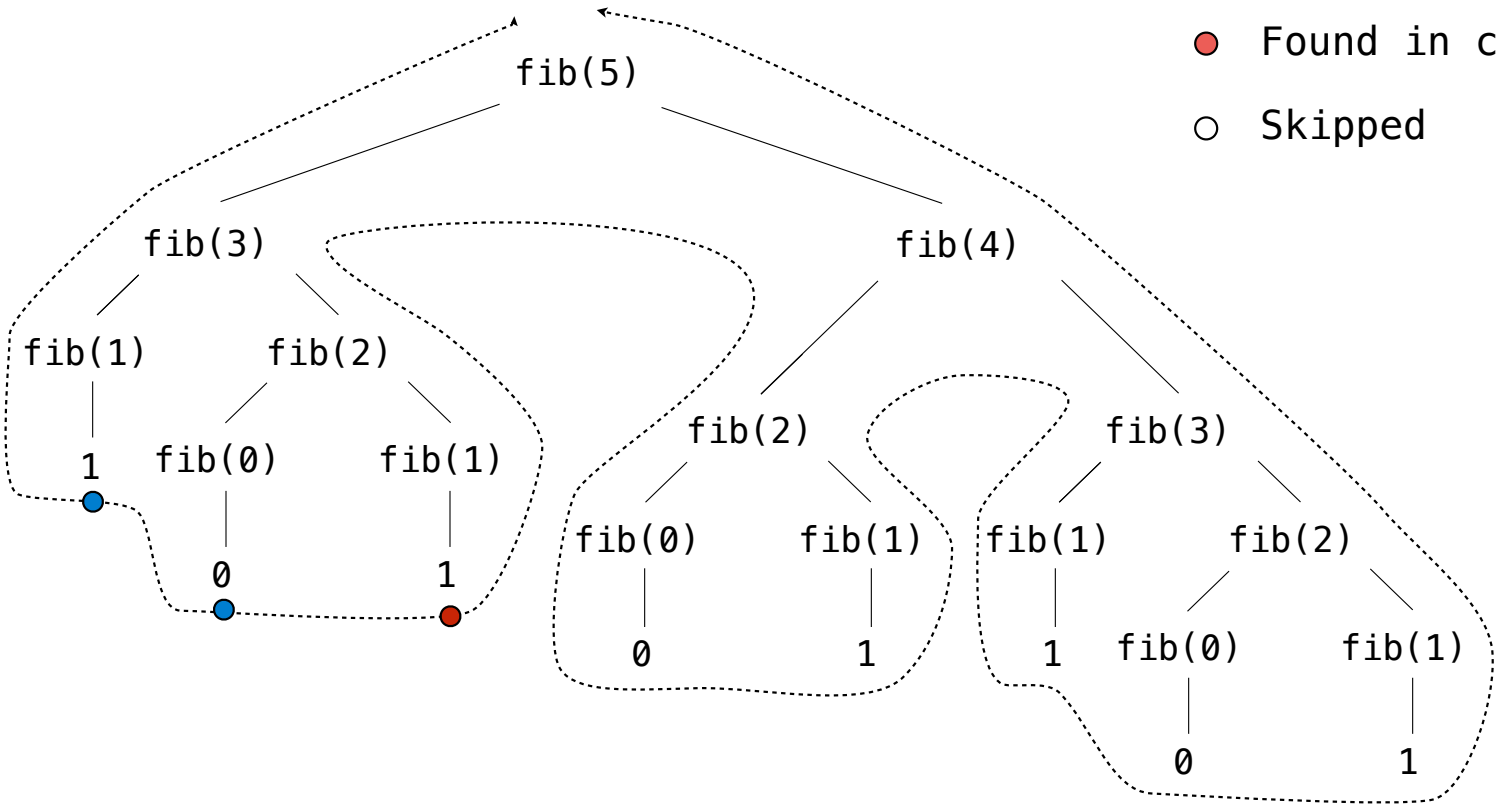
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



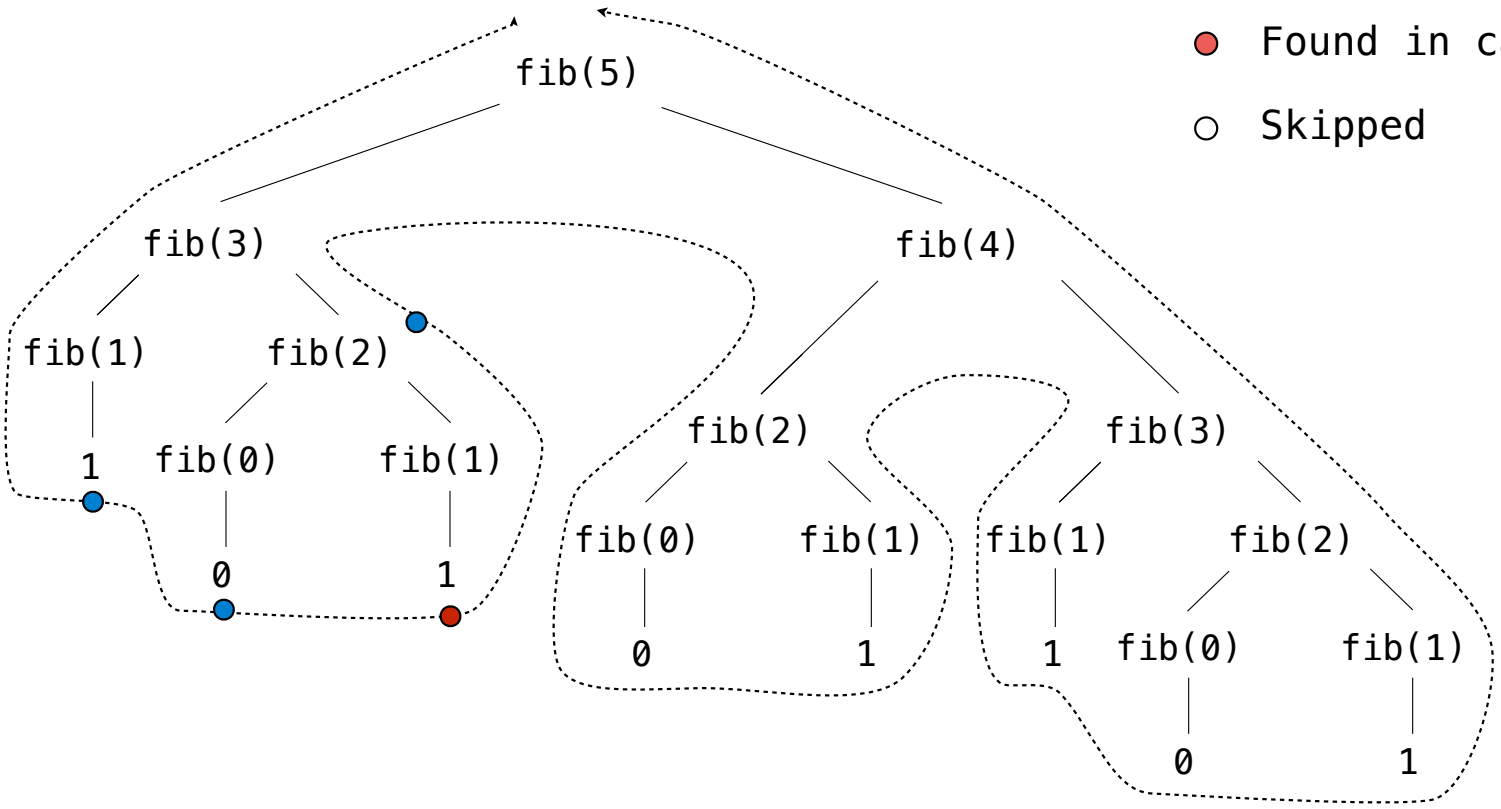
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



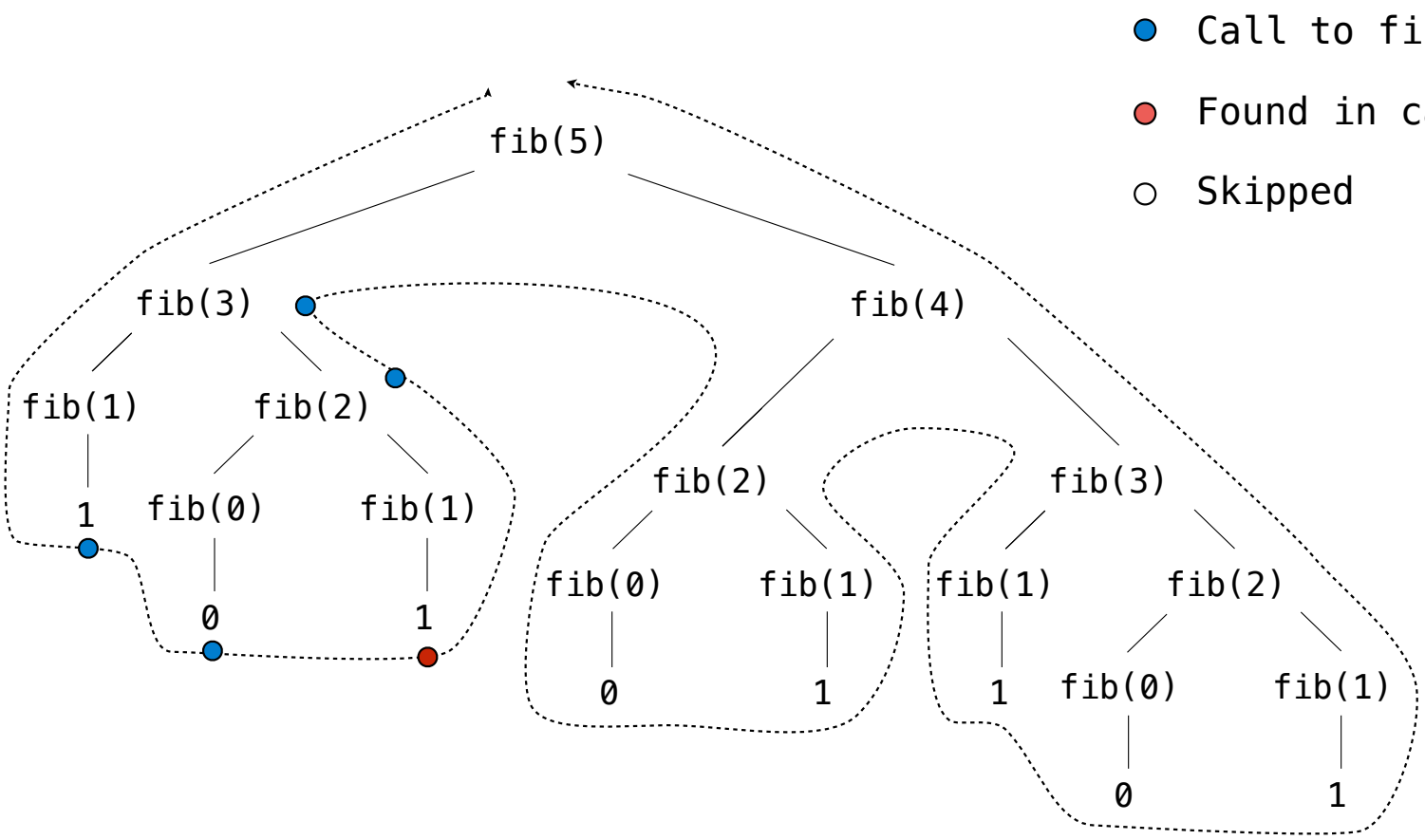
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



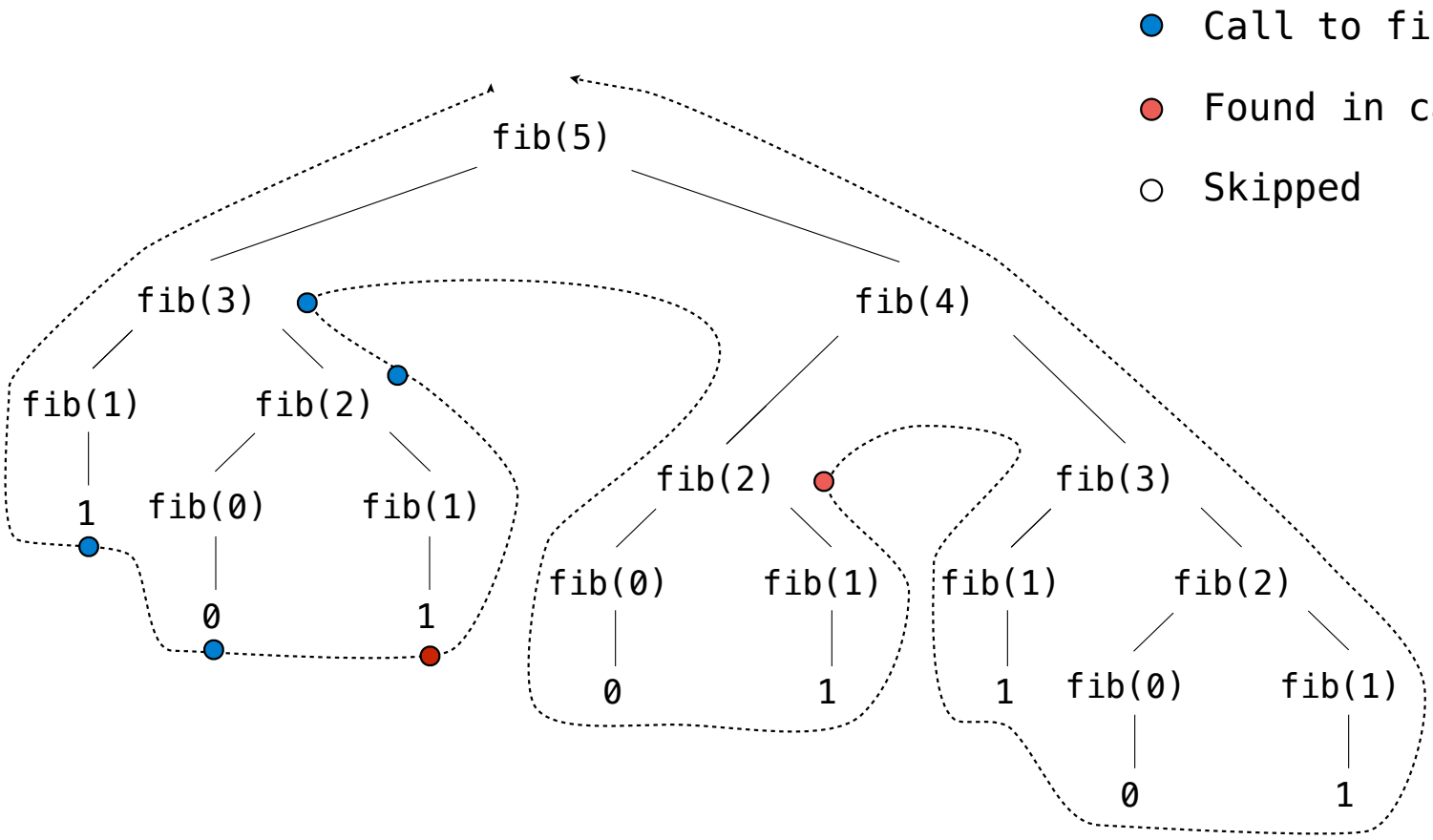
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



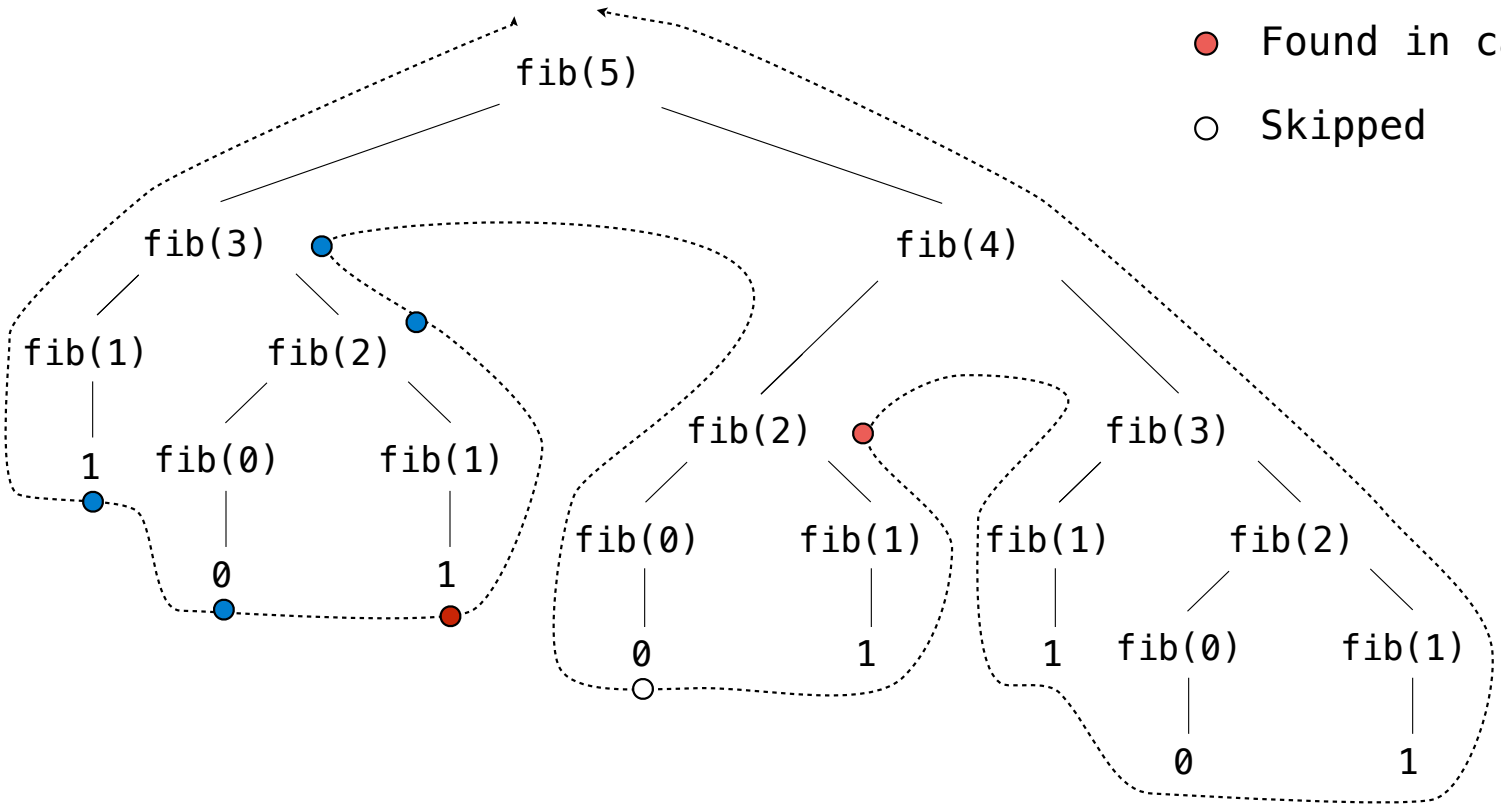
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



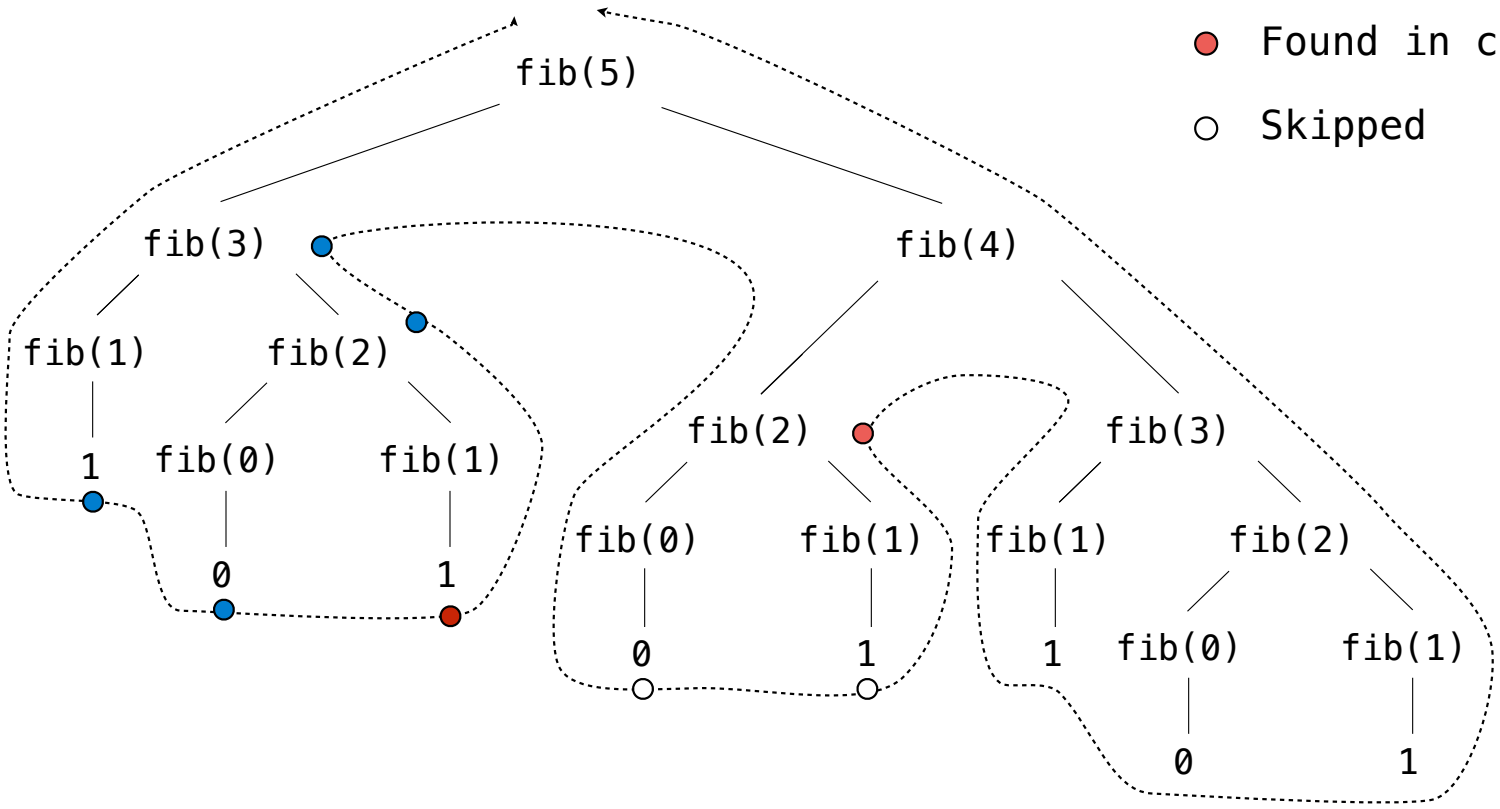
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



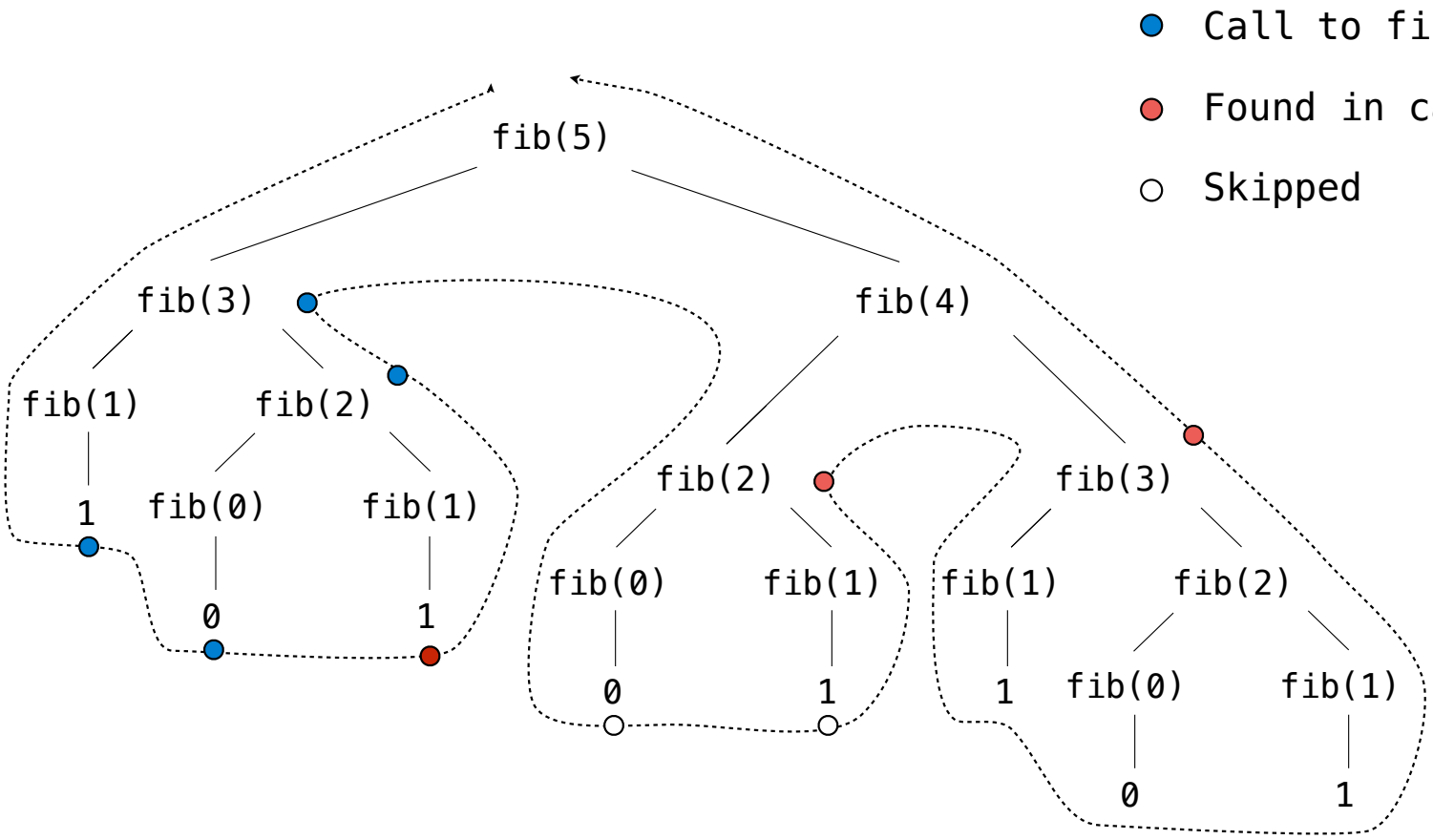
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



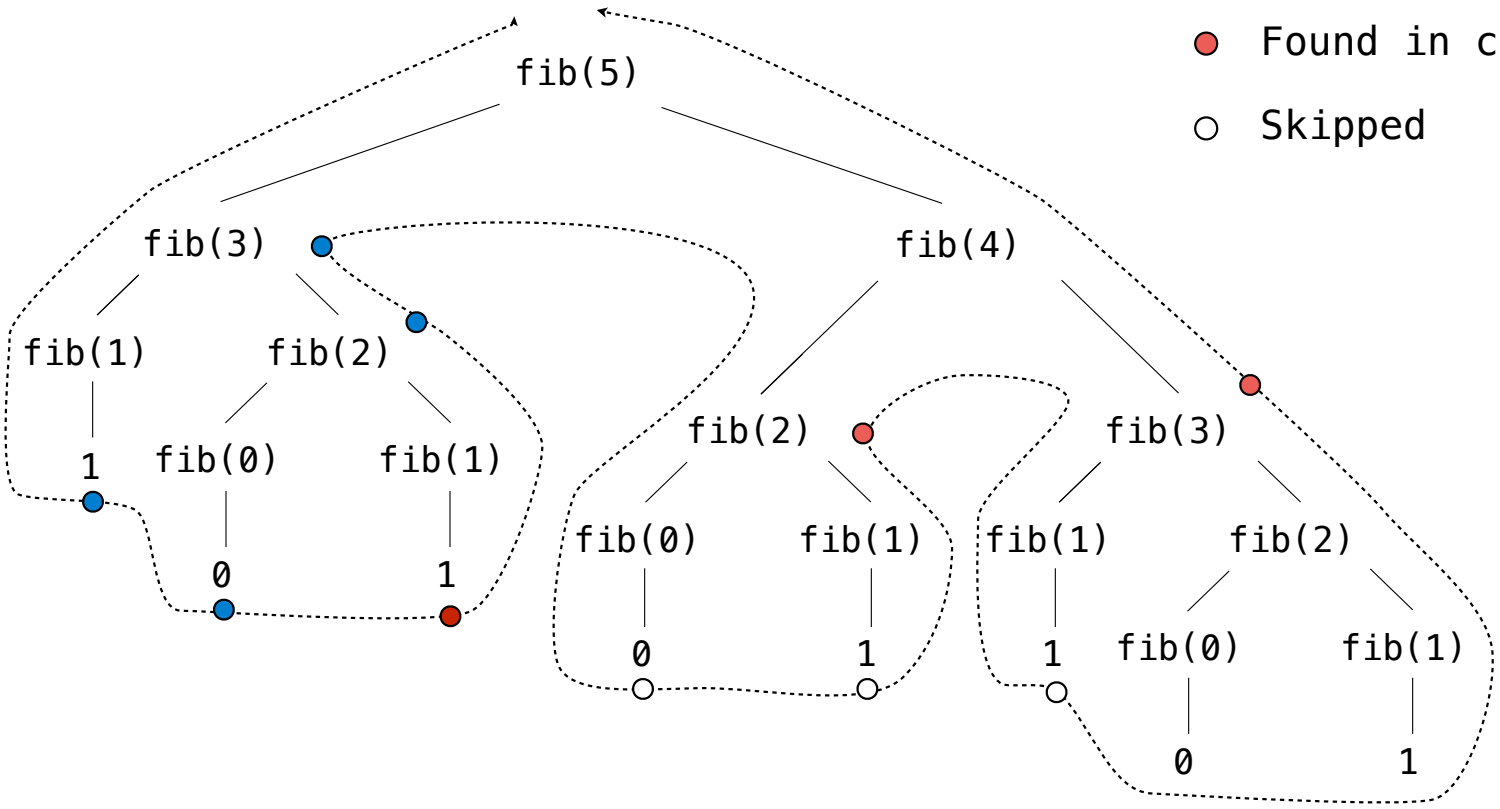
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



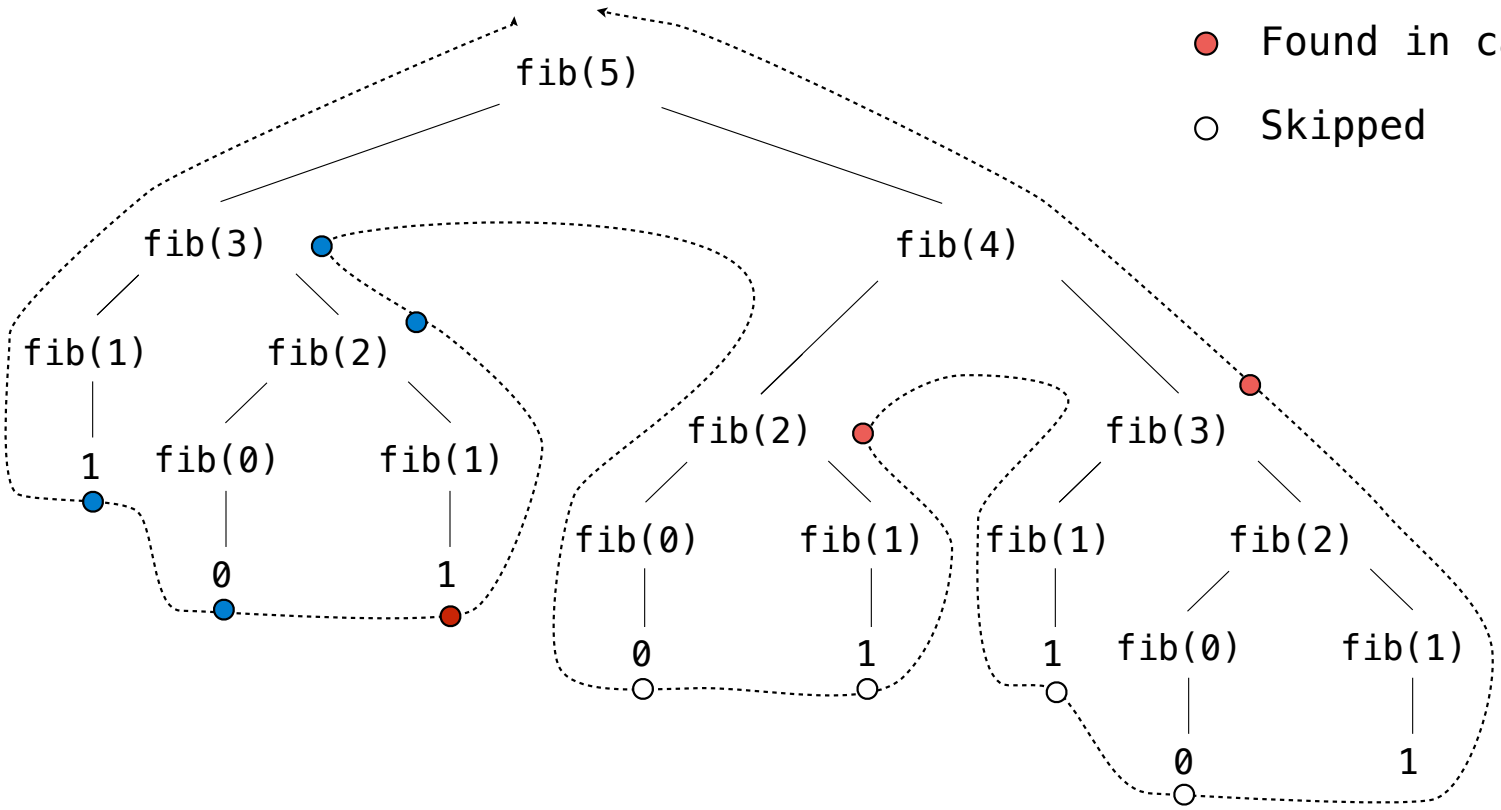
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



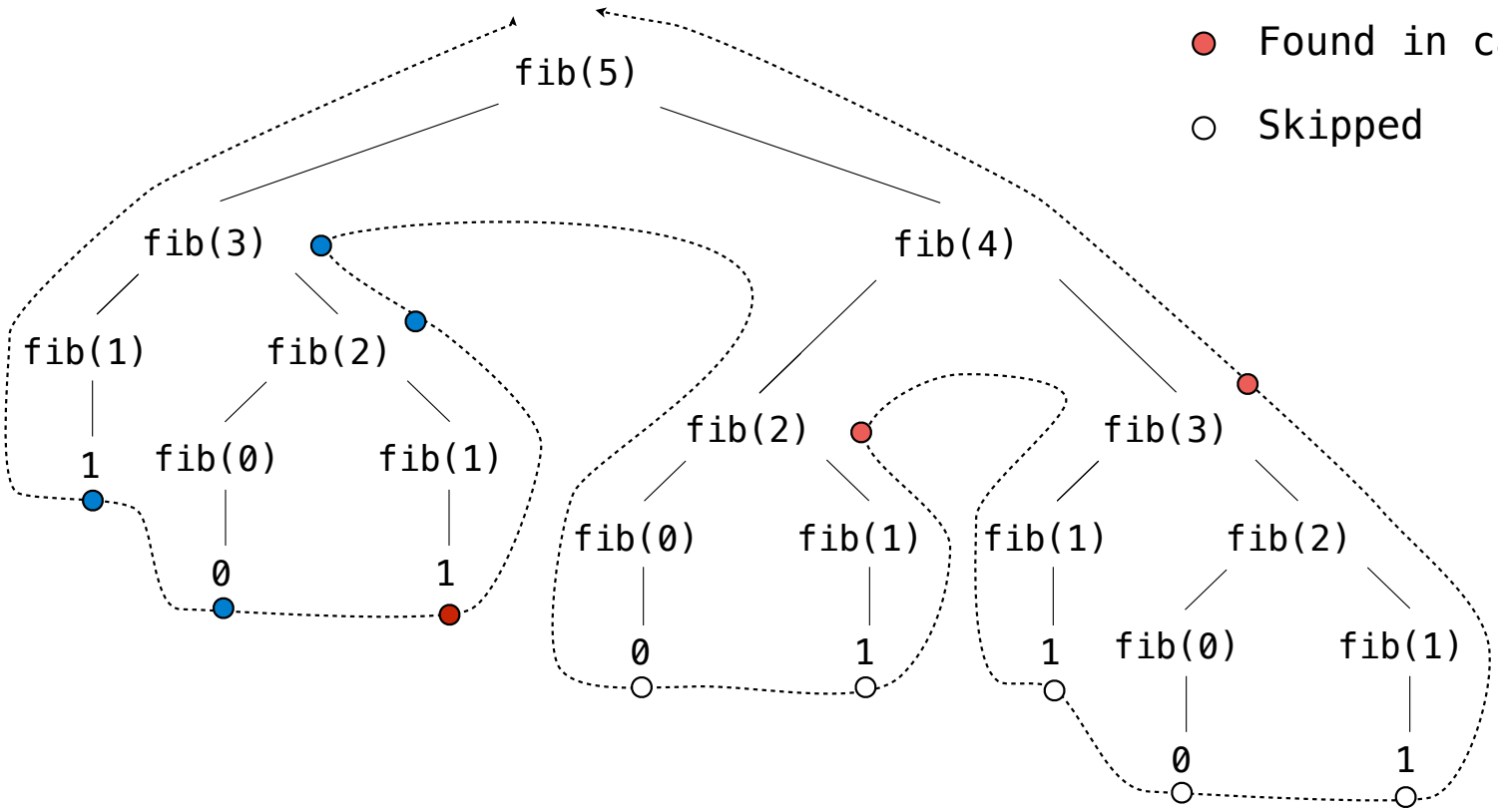
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



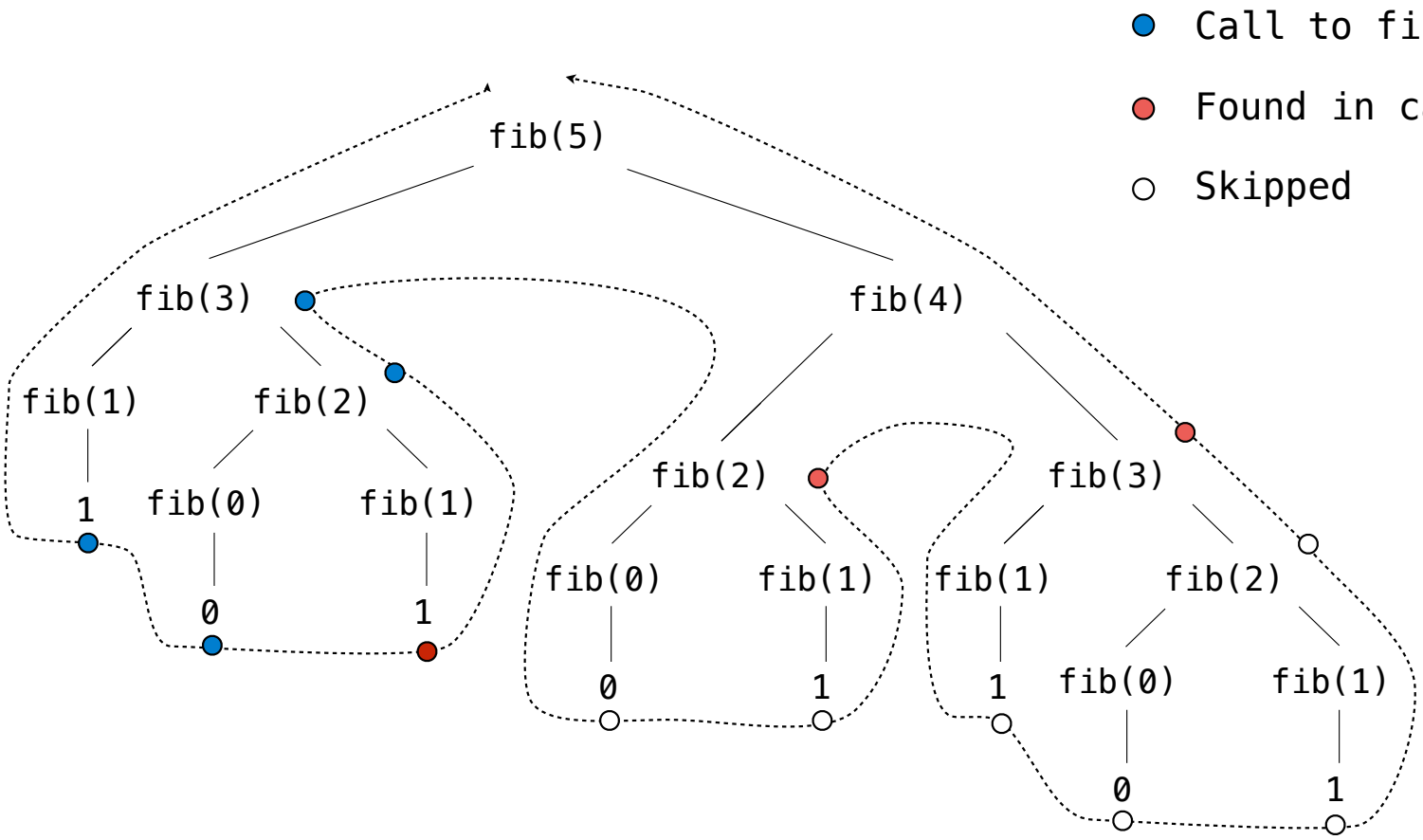
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



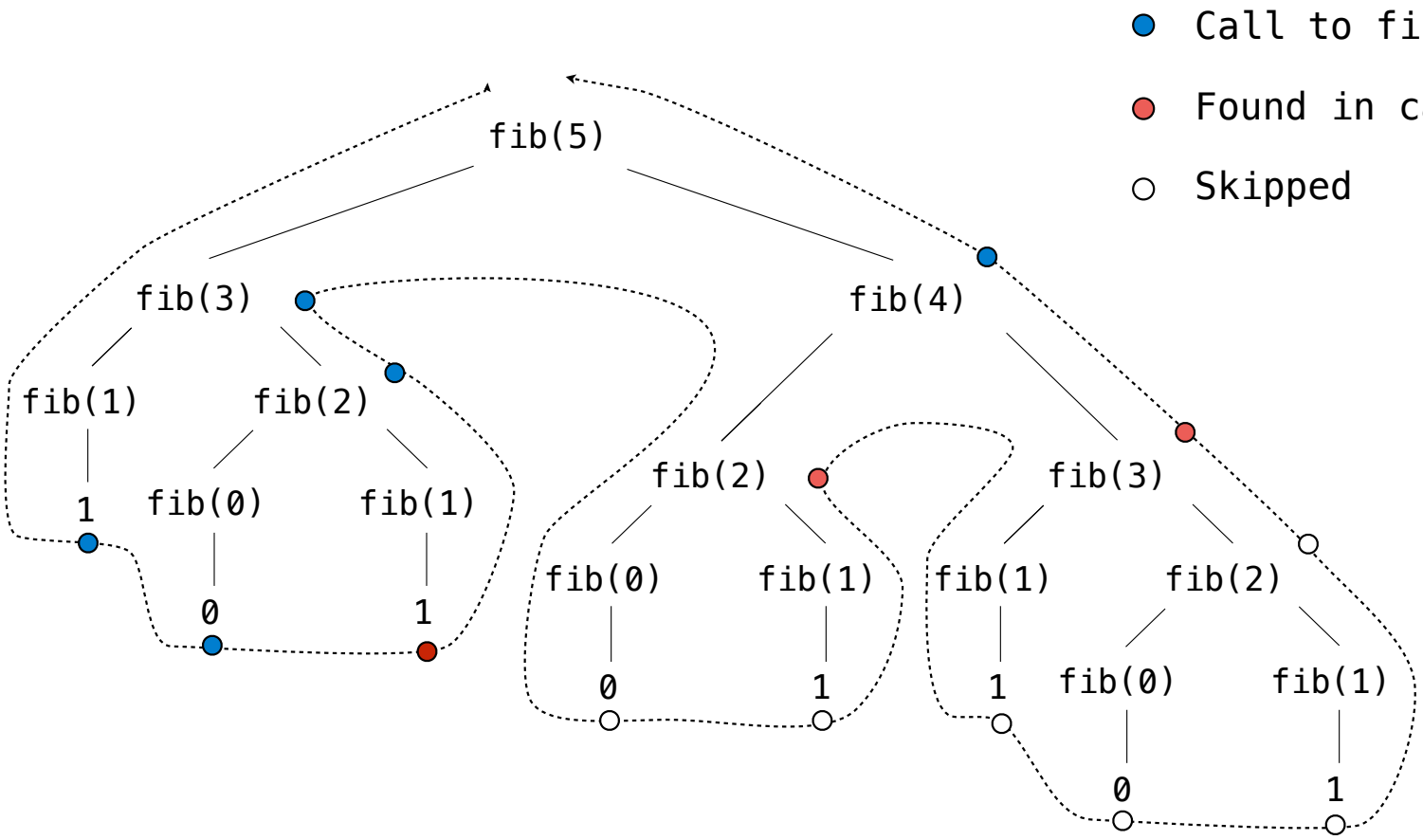
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



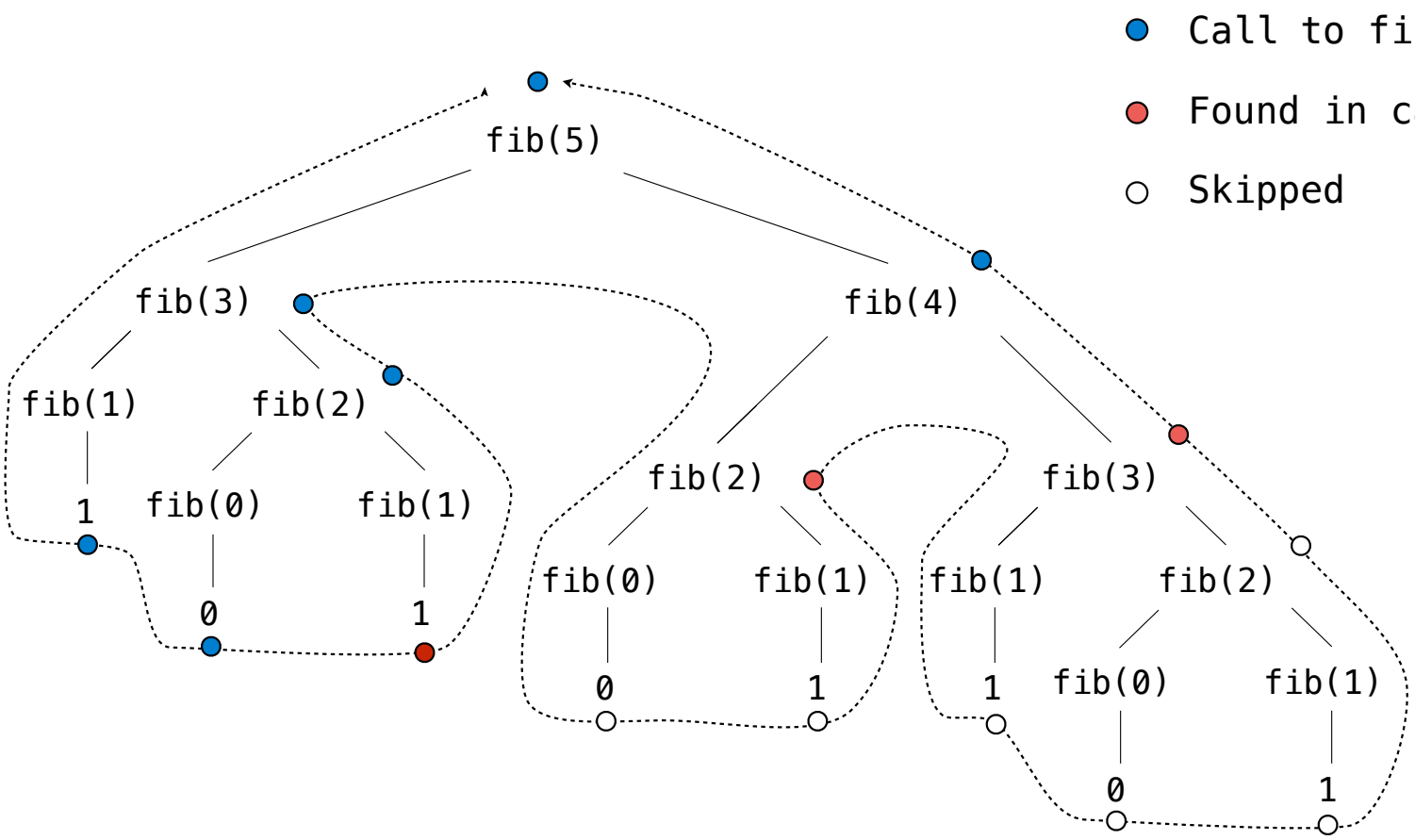
http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Memoized Tree Recursion



http://en.wikipedia.org/wiki/File:Fibonacci.jpg

Twenty-One (Nim)

Twenty-One Rules

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad

21

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad **21** **20**

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad

21 ⟵ **20**

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

19

Some states are good; some are bad

21 ⟵ **20**

Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

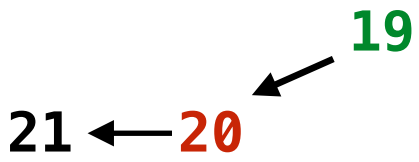
The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad



Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

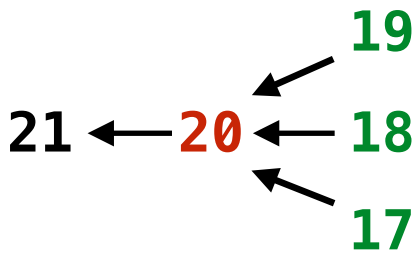
The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad



Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

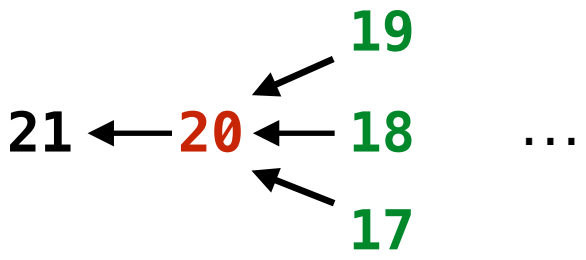
The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad



Twenty-One Rules

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

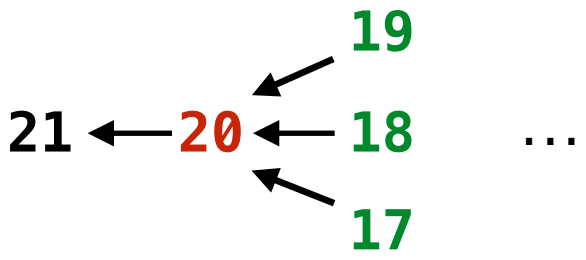
The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

(Demo)

Some states are good; some are bad



(Demo)

Hog Optimal Strategies

Church Numerals (Homework 2 Challenge Question)