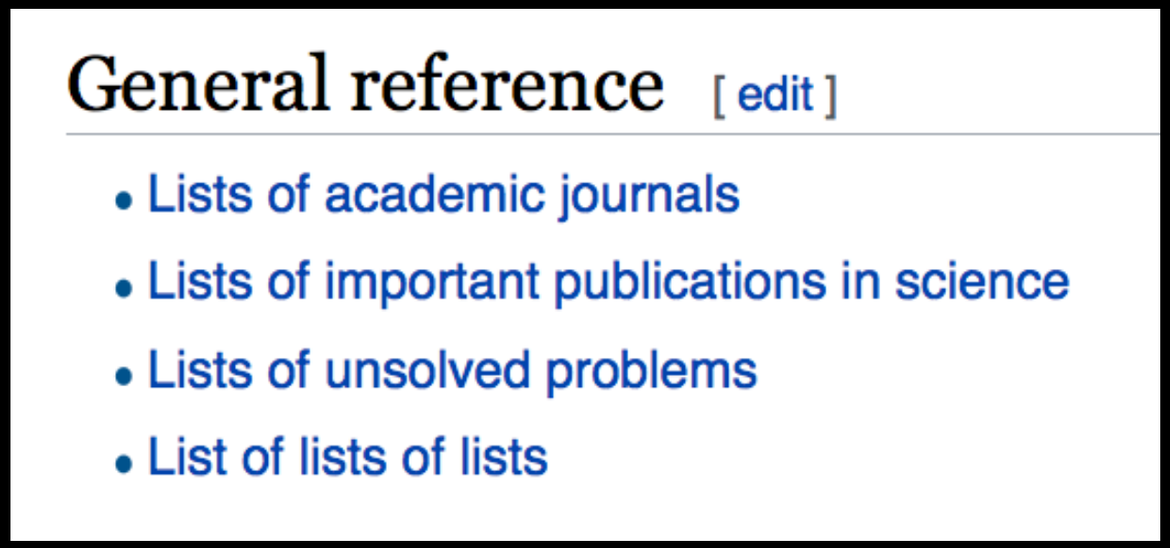
#2 (More) Environments and Recursion

TA: Jerry Chen (jerry.c@berkeley.edu)



General reference [edit]

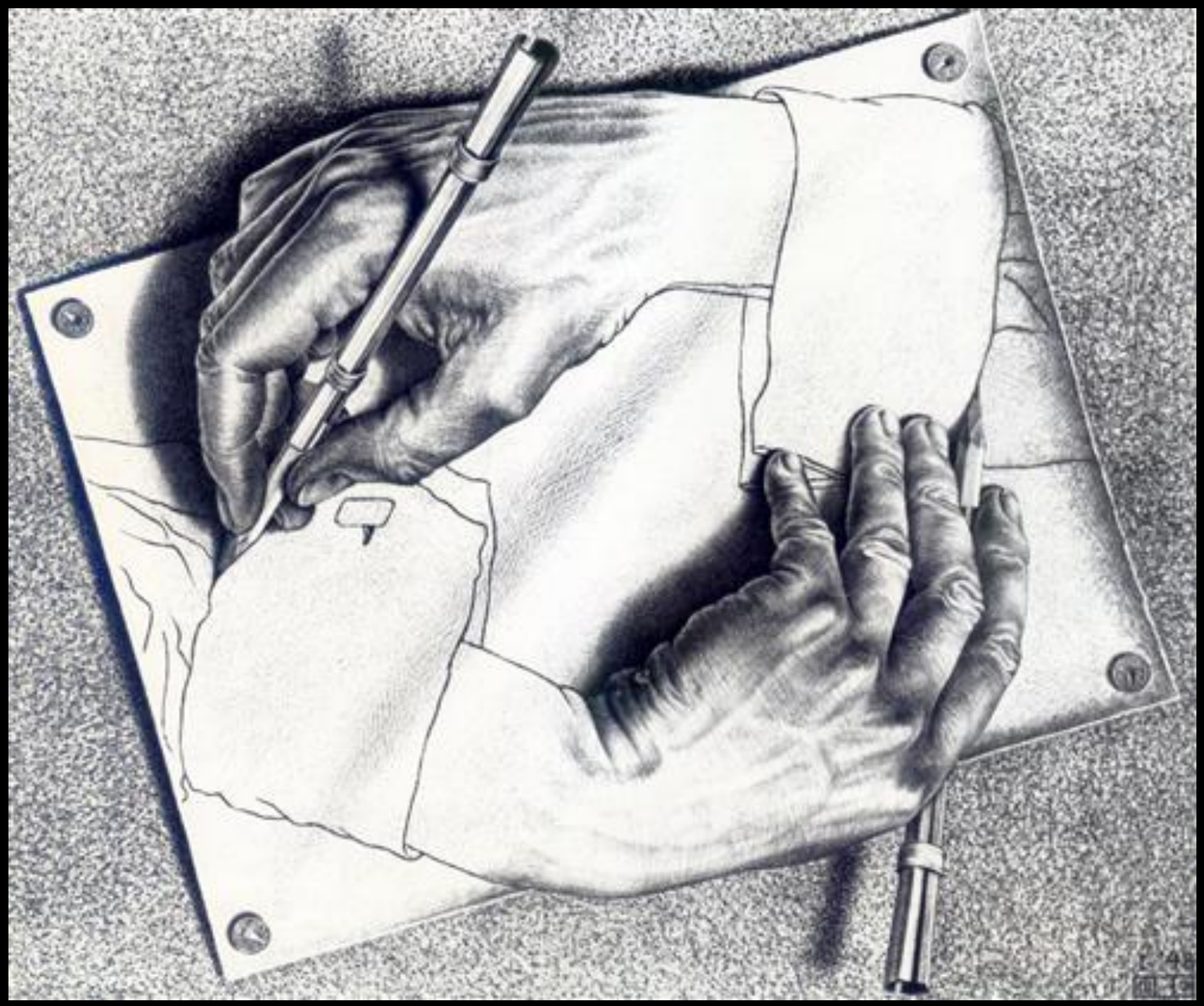
- Lists of academic journals
- Lists of important publications in science
- Lists of unsolved problems
- List of lists of lists

List of lists of lists from Wikipedia

Feedback

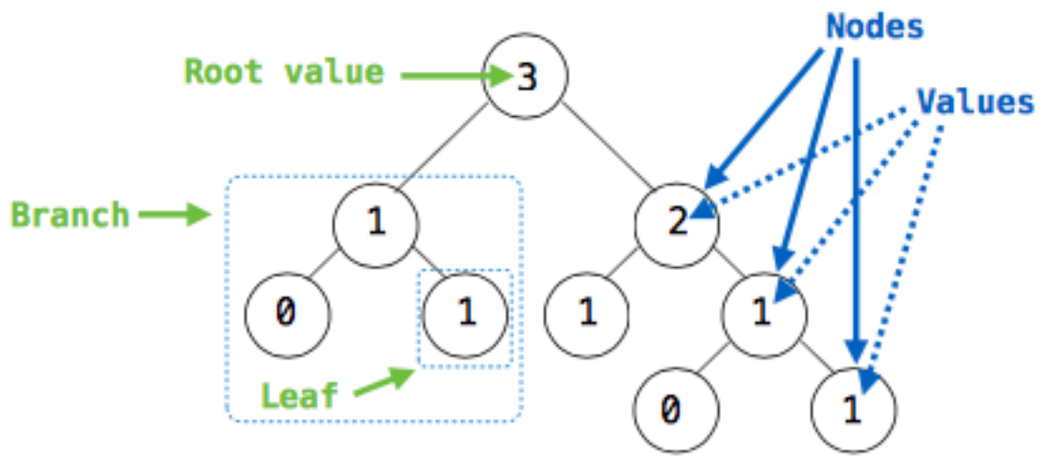
- Going too fast
- Going too slow
- Going *juuuust* right
- Quiz solutions?

Recursion



Drawing Hands by M. C. Escher

Tree Abstraction



Recursive description (**wooden trees**):

A **tree** has a **root** value and a list of **branches**

Each branch is a **tree**

A tree with zero branches is called a **leaf**

Relative description (**family trees**):

Each location in a tree is called a **node**

Each **node** has a **value**

One node can be the **parent/child** of another

People often refer to values by their locations: "each parent is the sum of its children"

Components of Recursion

3 Easy Steps

1. Solve **base case**
2. **Recursive call** on a subproblem
3. **Use the result** to solve the original problem

```python
def factorial(n):
    if n == 0:
        return 1
    return n * factorial(n - 1)
```

```python
def factorial(n):
    if n == 0:
        return 1
    return n * factorial(n - 1)
```

```
1 def factorial(n):
2     if n == 0:
3         return 1
4     return n * factorial(n - 1)
```

```python
def factorial(n):
    if n == 0:
        return 1
    return n * factorial(n - 1)
```

```
1 def hailstone(n):
2     print(n)
3     if n == 1:
4         return
5     elif n % 2 == 0:
6         hailstone(n - 1)
7     else:
8         hailstone(n - 1)
```

What's wrong?

```
1 def hailstone(n):
2     print(n)
3     if n == 1:
4         return
5     elif n % 2 == 0:
6         hailstone(n - 1)
7     else:
8         hailstone(n - 1)
```

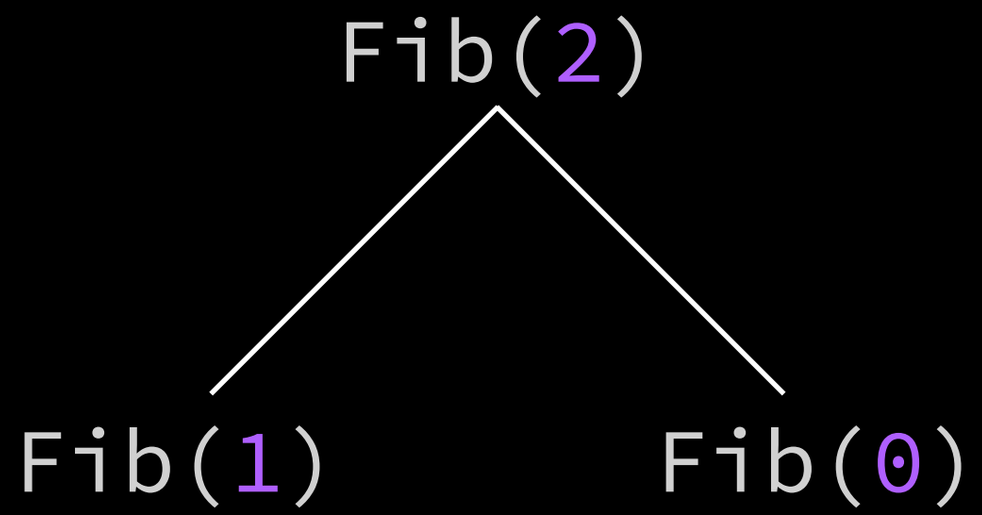
Tree Recursion

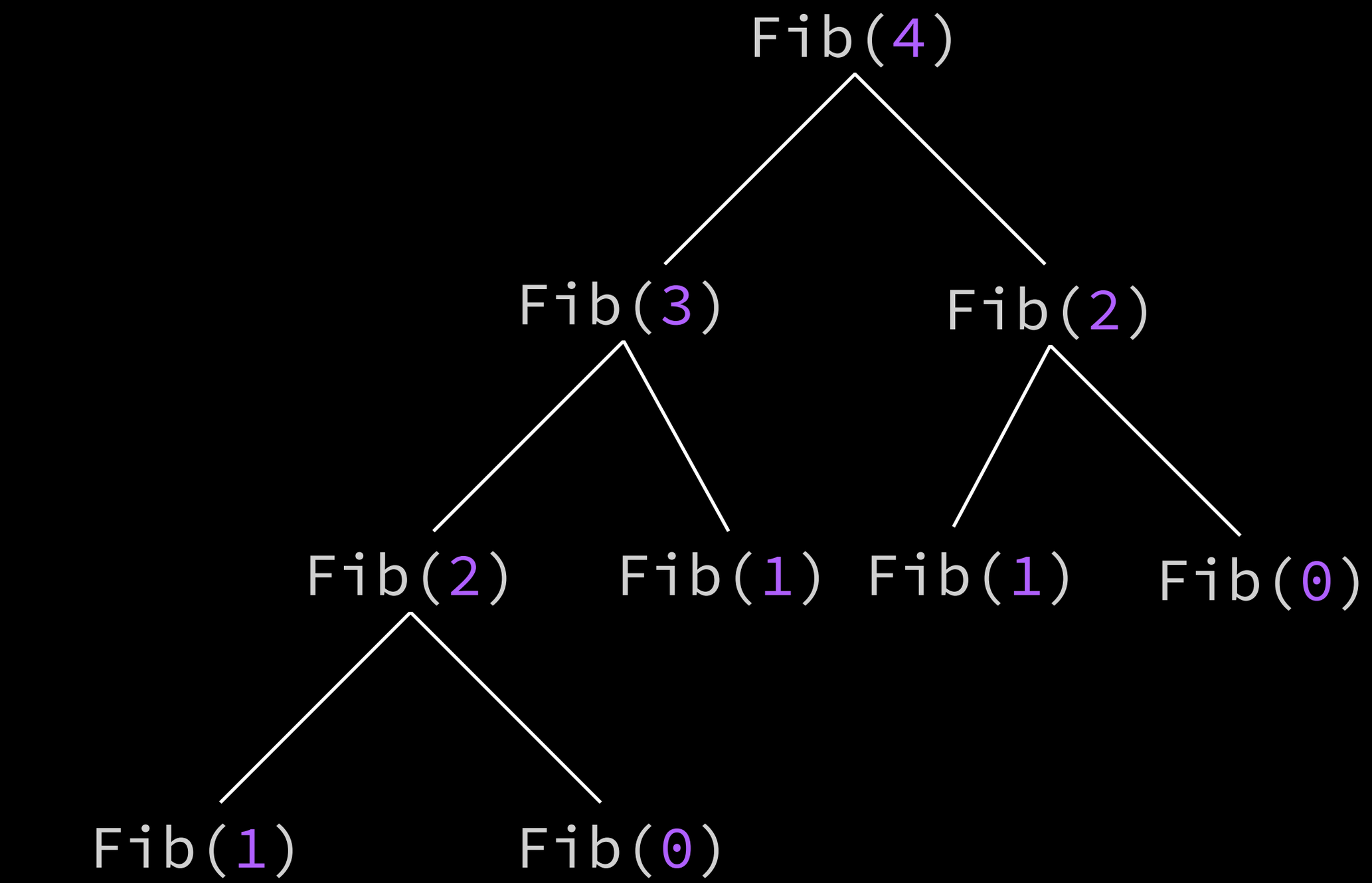
Call **multiple** functions

Useful for representing choices

```
Fib(n) = Fib(n - 1) + Fib(n - 2)
```

```
Fib(2) = Fib(1) + Fib(0)
```


Fib(2)
Fib(1)
Fib(0)


Fib(4)
Fib(3)
Fib(2)
Fib(2)
Fib(1)
Fib(1)
Fib(0)
Fib(1)
Fib(0)