Dynamic Programming

http://bit.ly/VTProgrammingDP

Dynamic Programming

A method for solving a complex problem by breaking it down into a collection of simpler subproblems, solving each only once

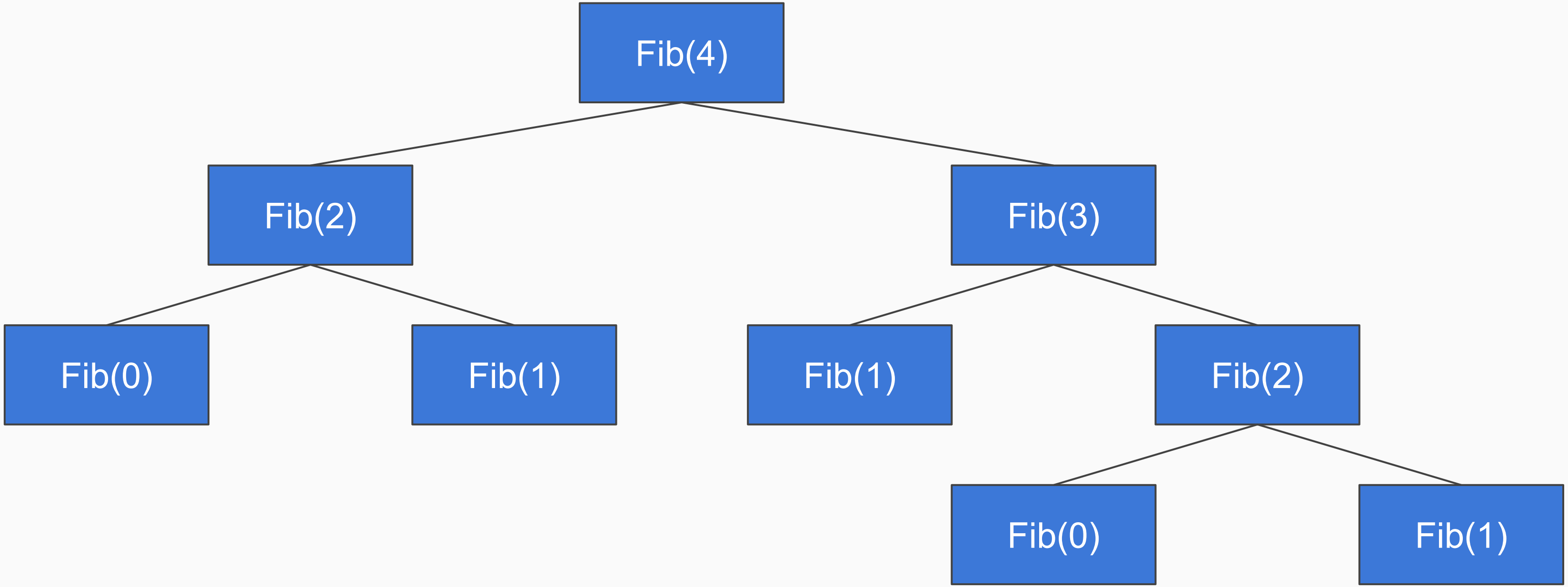
Examples

- Making change
- Path finding
- Packing a bag

Parallels to recursion

- Recursion uses a similar idea of breaking a problem down into subproblems and using those solutions

Fibonacci



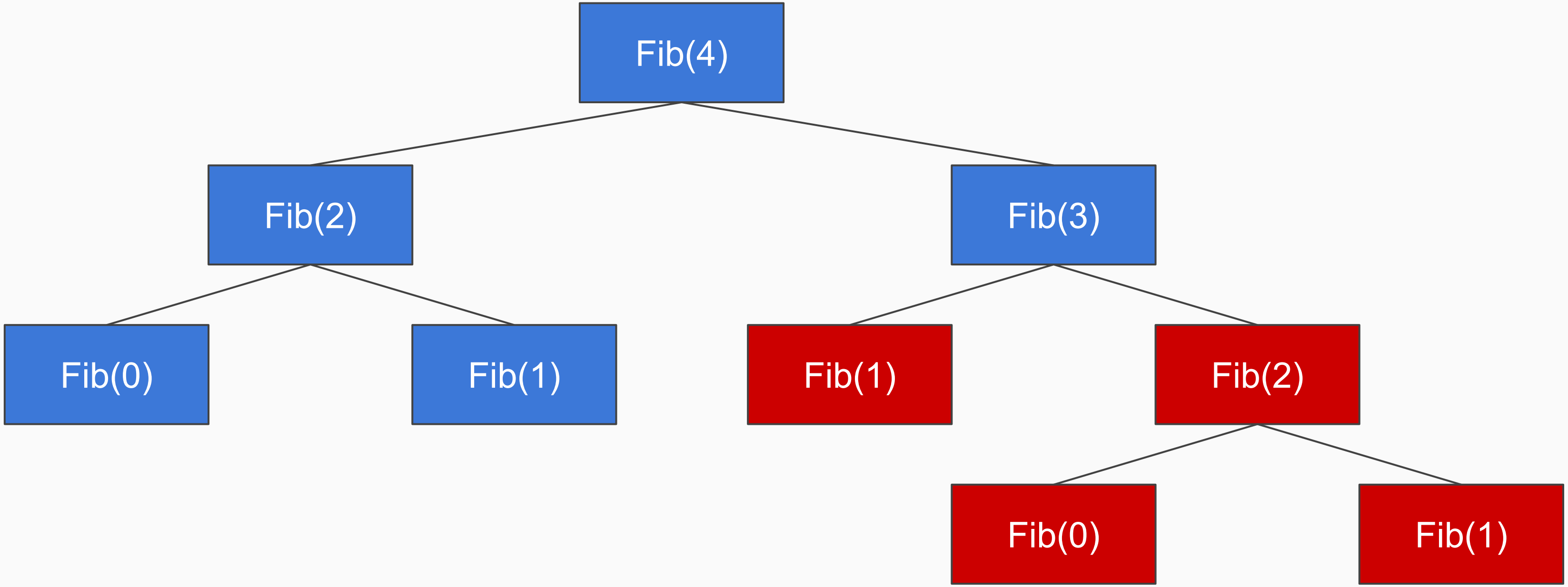
Code

```
public int fib(int N) {
   if (N == 0 || N == 1) {
       return 1;
   } else {
       return fib(N - 2) + fib(N - 1);
   }
}
```

Top-down DP

Start at the highest level problem, then decompose it into smaller problems as you need.

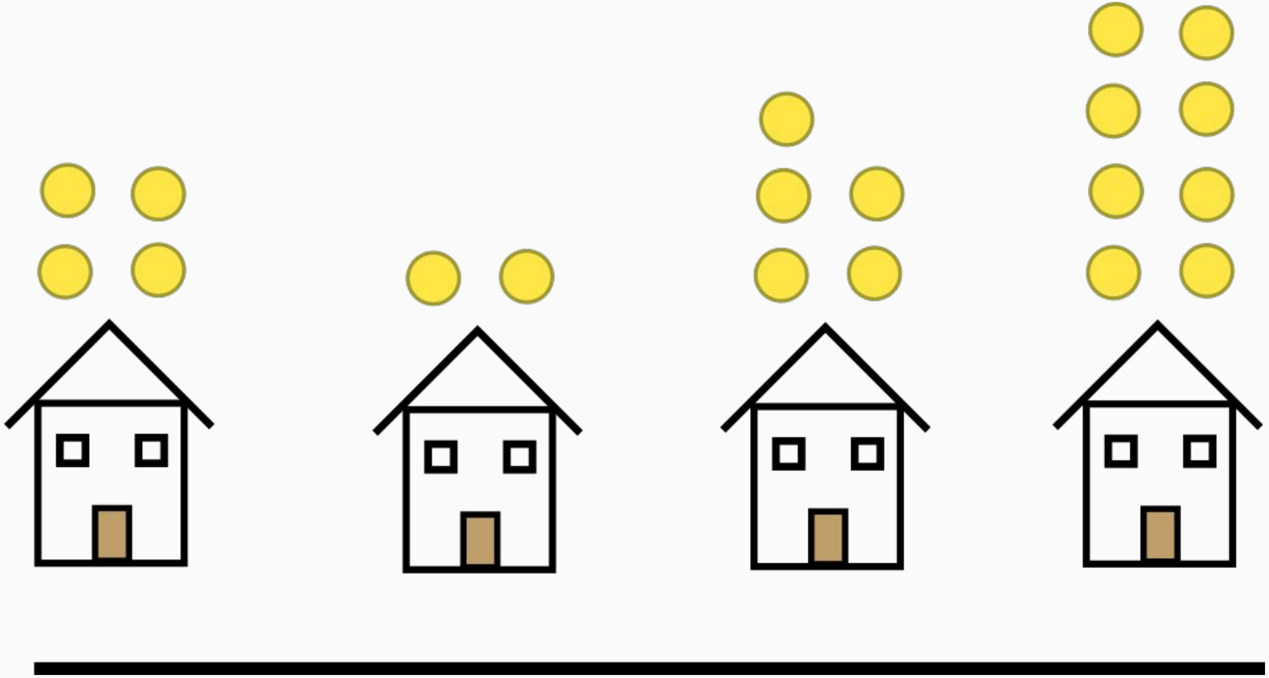
Fibonacci - DP Top Down



Bottom-Up DP

Solve the smaller subproblems, then build up to the solution you are looking for.

Example problem



```
static int A = [4,2,5,8];
static int M(int i) {
    if (i >= A.length) {
        return 0;
    }
    return Math.max(
        A[i] + M(i + 2),
        M(i + 1),
    );
}
```

```
static int A = [4,2,5,8];
static Integer[] dp = new Integer[N];
static int M(int i) {
    if (i >= A.length) {
        return 0;
    }
    if (dp[i] == null) {
        dp[i] = Math.max(
             A[i] + M(i + 2),
             M(i + 1),
        );
    }
    return dp[i];
}
```

Problems

- https://projecteuler.net/problem=67
- https://projecteuler.net/problem=31
- https://projecteuler.net/problem=15

More problems

- https://open.kattis.com/problems/welcomehard
- https://open.kattis.com/problems/countcircuits
- https://open.kattis.com/problems/tractor
    - Can be solved with clever DP or with some CS-based math

More problems

- https://icpcarchive.ecs.baylor.edu/index.php?option=com_onlinejudge&Itemid=8&category=618&page=show_problem&problem=4502
- https://icpcarchive.ecs.baylor.edu/index.php?option=com_onlinejudge&Itemid=8&category=534&page=show_problem&problem=3956
    - Rebuilding paths from DP

Massive input for problem 67

- https://spruett.me/blog/static/code/huge-pyramid.java.html