DYNAMIC PROGRAMMING: ROD CUTTING

Jon Francis Pasana
Ralph Eduard So
Jameson Adriel Perez
Asherrie Jaye Tan

DYNAMIC PROGRAMMING? WHAT'S THAT?

Simply put: optimization of a recursive problem/solution.

If we have a recursive solution that has repeated calls for similar inputs, dynamic programming can be used to optimize it. You can break down the problem into sub-problems and store results, so you don't have to re-compute them later. It's better to see an example!

FIBONACCI SEQUENCE

The Fibonacci Sequence

1,1,2,3,5,8,13,21,34,55,89,144,233,377…

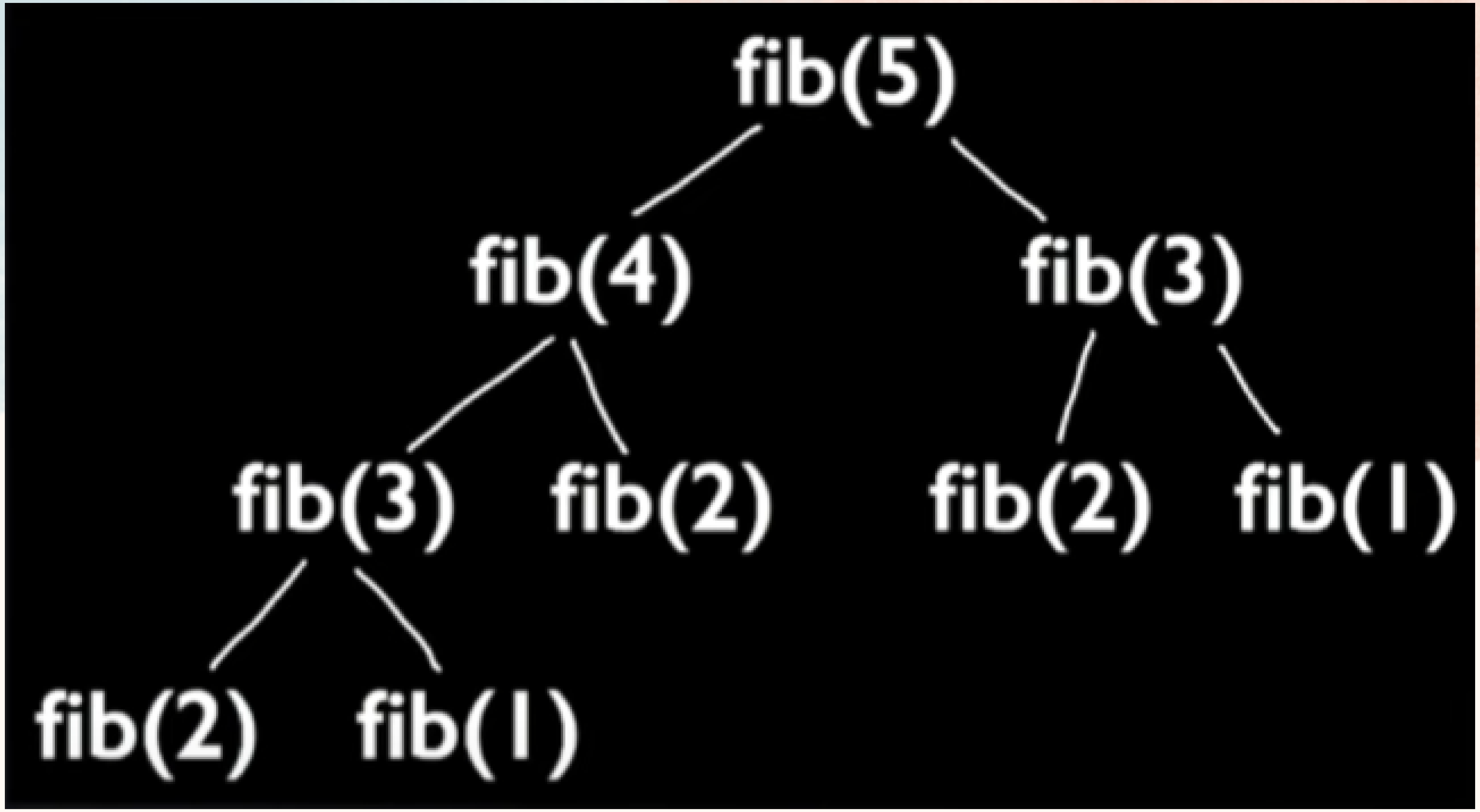
1+1=2	13+21=34
1+2=3	21+34=55
2+3=5	34+55=89
3+5=8	55+89=144
5+8=13	89+144=233
8+13=21	144+233=377

FIBONACCI SEQUENCE

pseudocode:

```
int fib(n)
    if n == 1 or n == 2
        result = 1
    else
        result = fib(n-1) + fib(n-2)
    return result
```

FIBONACCI SEQUENCE



Time complexity: Exponential

$O(n^2)$

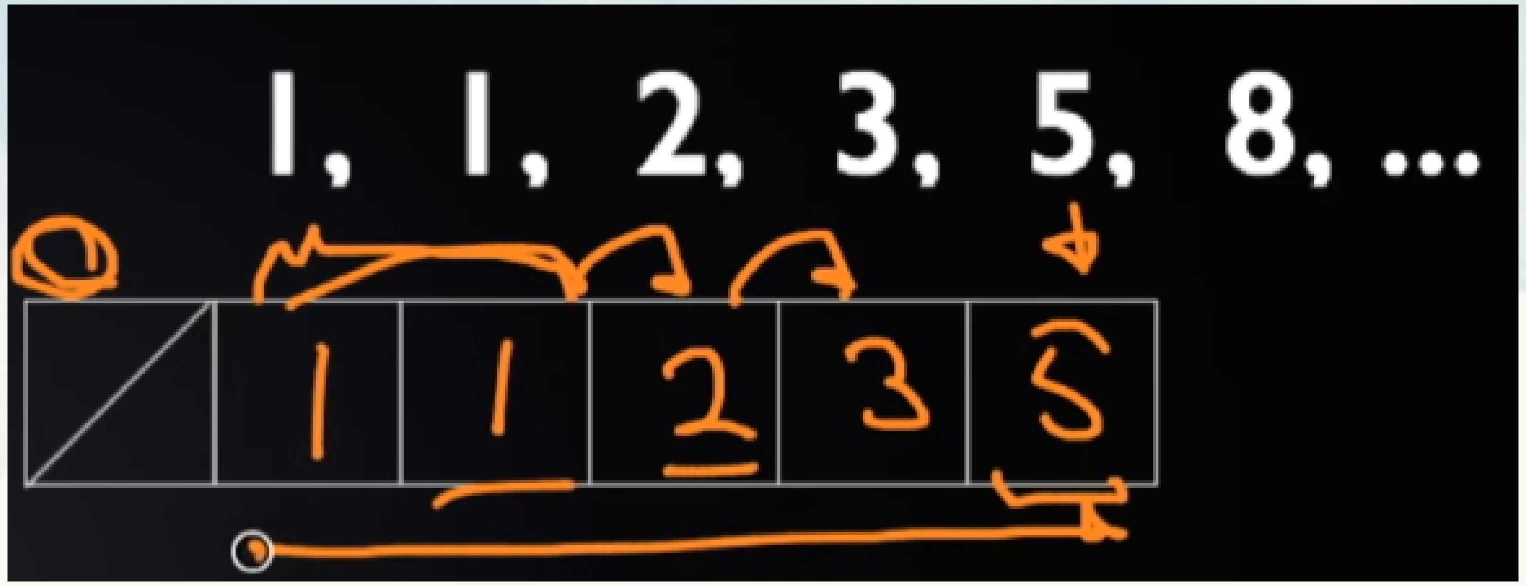
FIBONACCI SEQUENCE

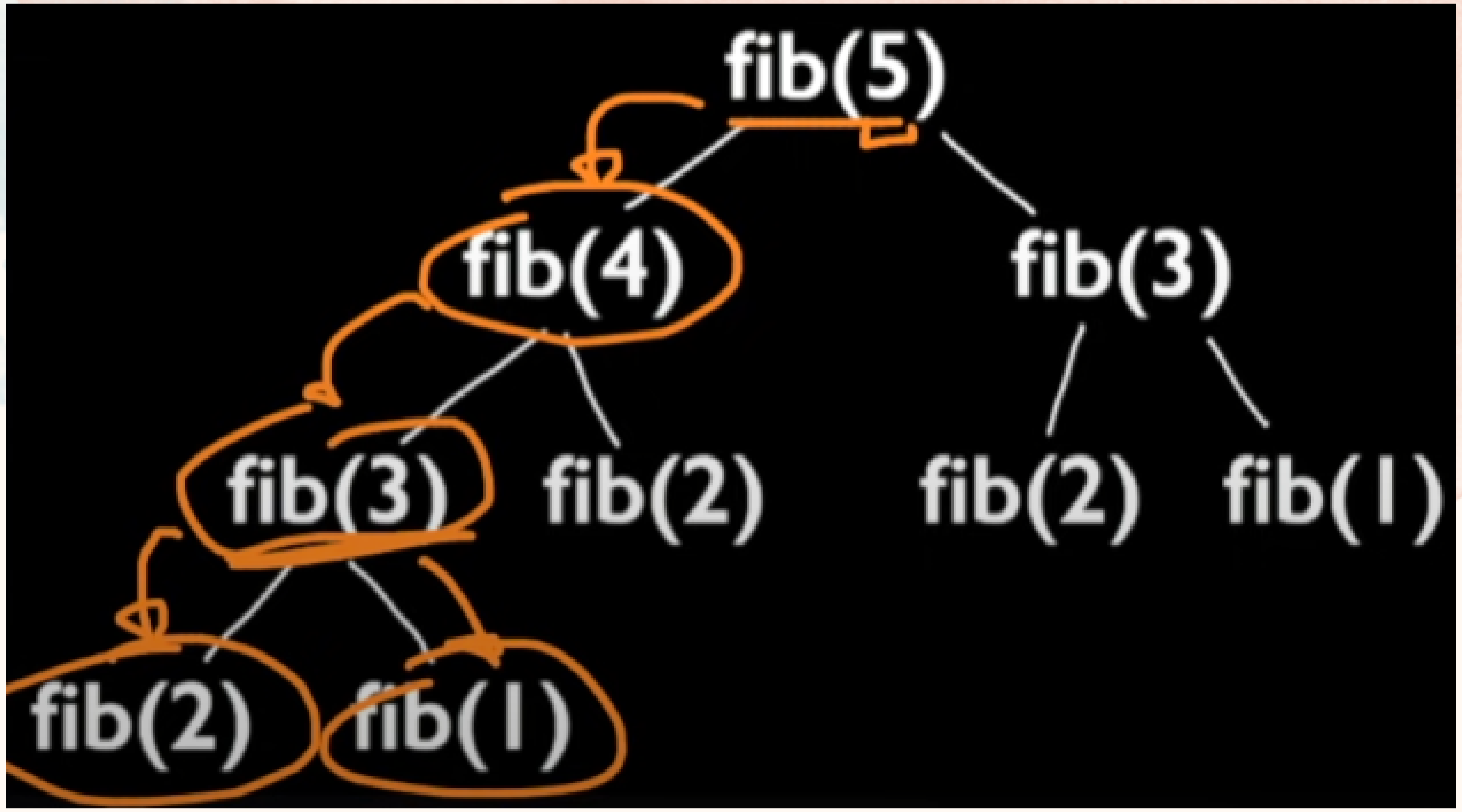
pseudocode:

```
int memo[100] /* create array before calling fib

int fib(n, memo)
    if memo[n] != null
        return memo[n]
    if n == 1 or n == 2
        result = 1
    else
        result = fib(n-1) + fib(n-2)
    memo[n] = result
    return result
```

FIBONACCI SEQUENCE

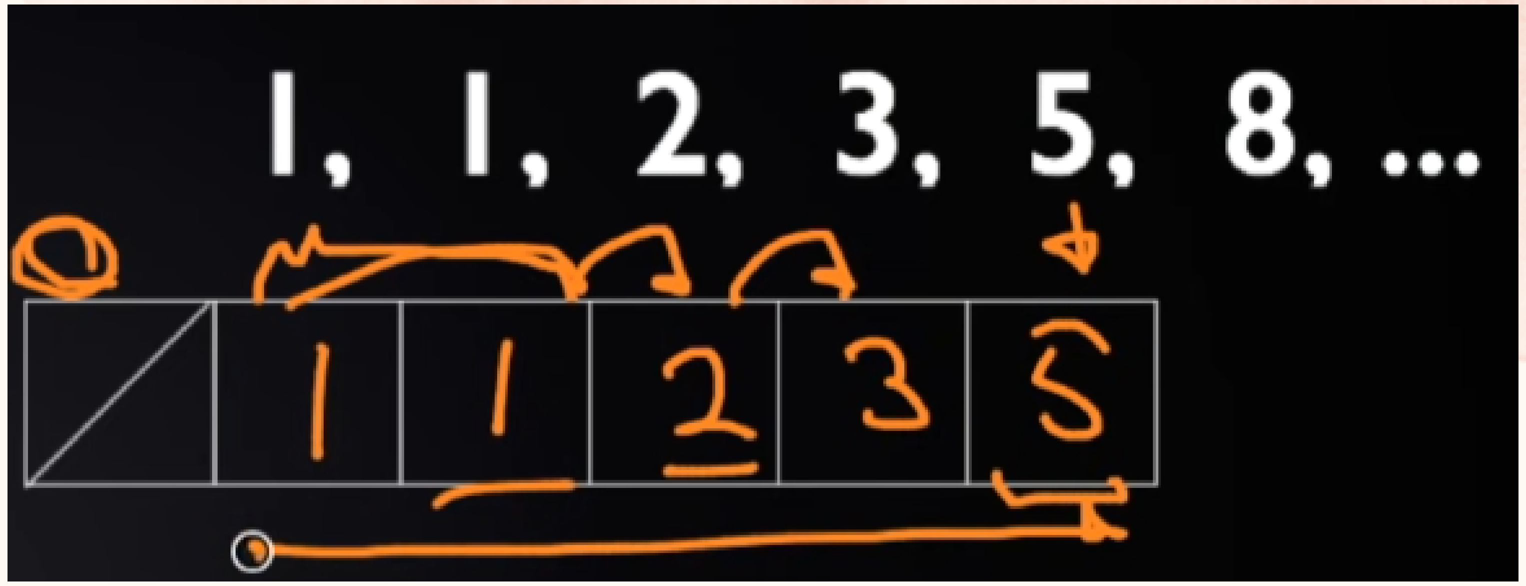




Time complexity: O(n)

FIBONACCI SEQUENCE

this method of dynamic programming - wherein you store answers to previously solved subproblems in an array or table - is called *Memoization*

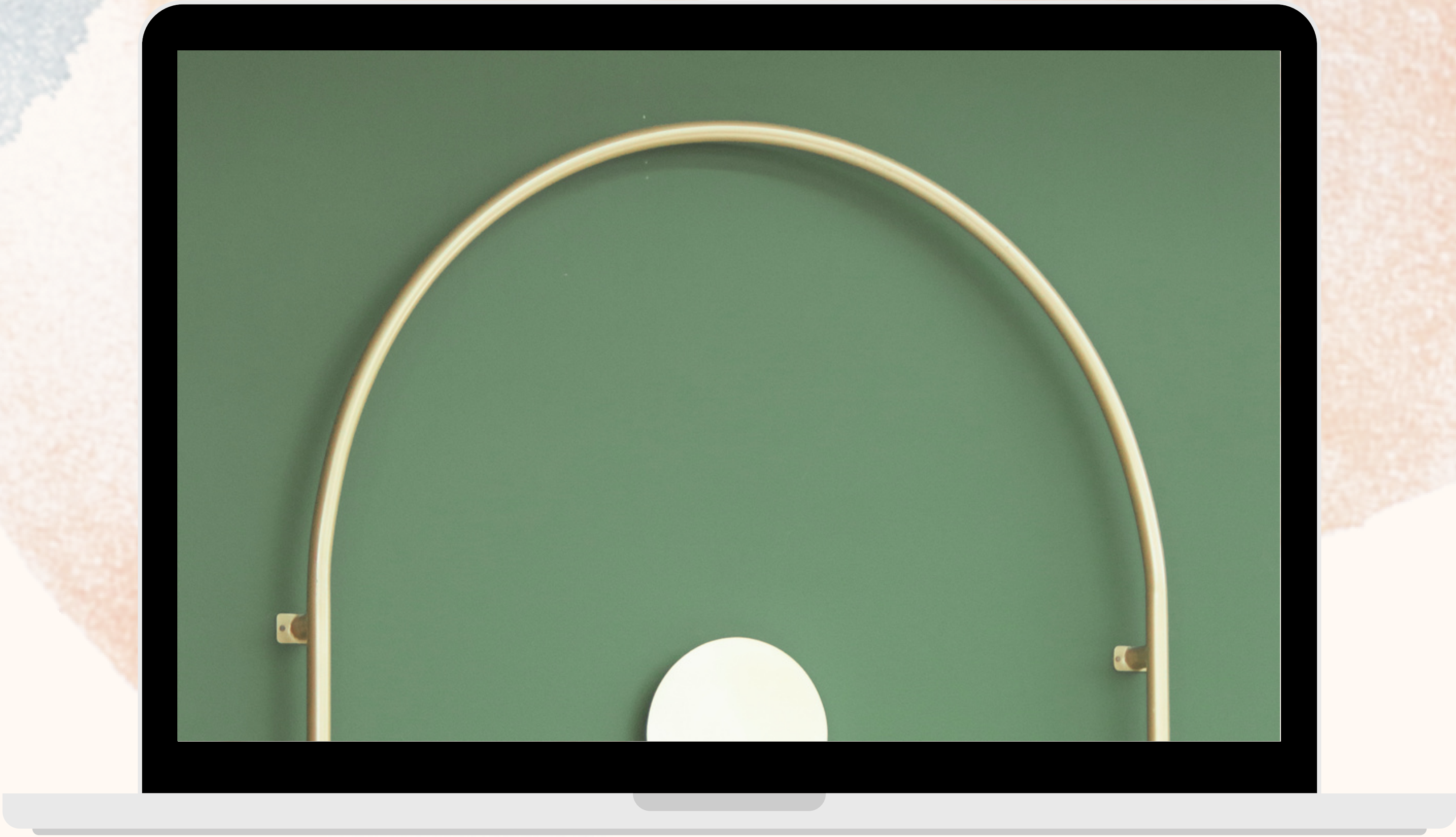


DEVELOPMENT of Dynamic Programming

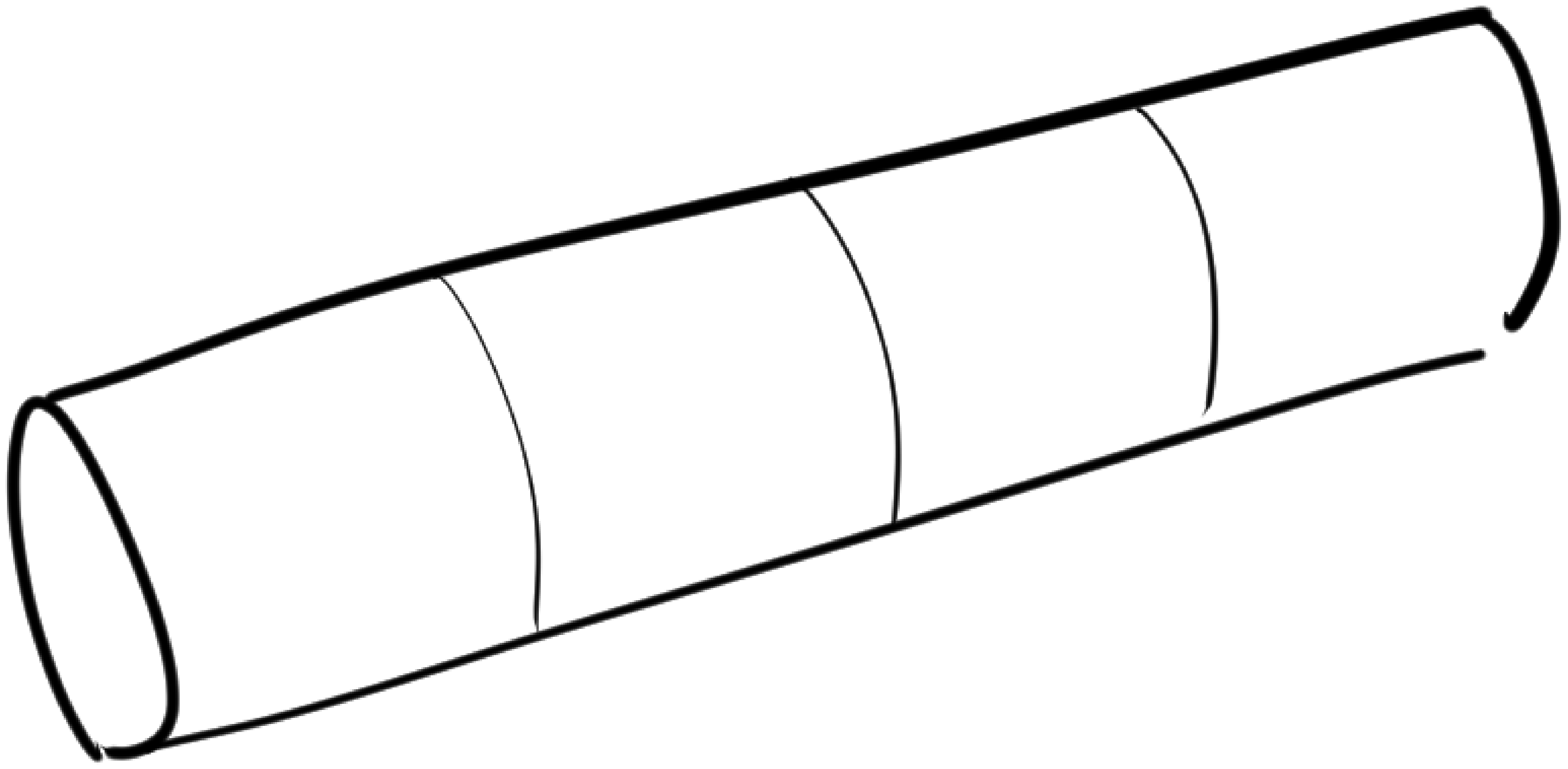
- It is a mathematical optimization method and a computer programming method.
- It was developed by Richard Bellman in the 1950's
- It has many application, from aerospace engineering to economics.

ORIGINAL VERSION

- Basically the original version of dynamic programming is recursion.

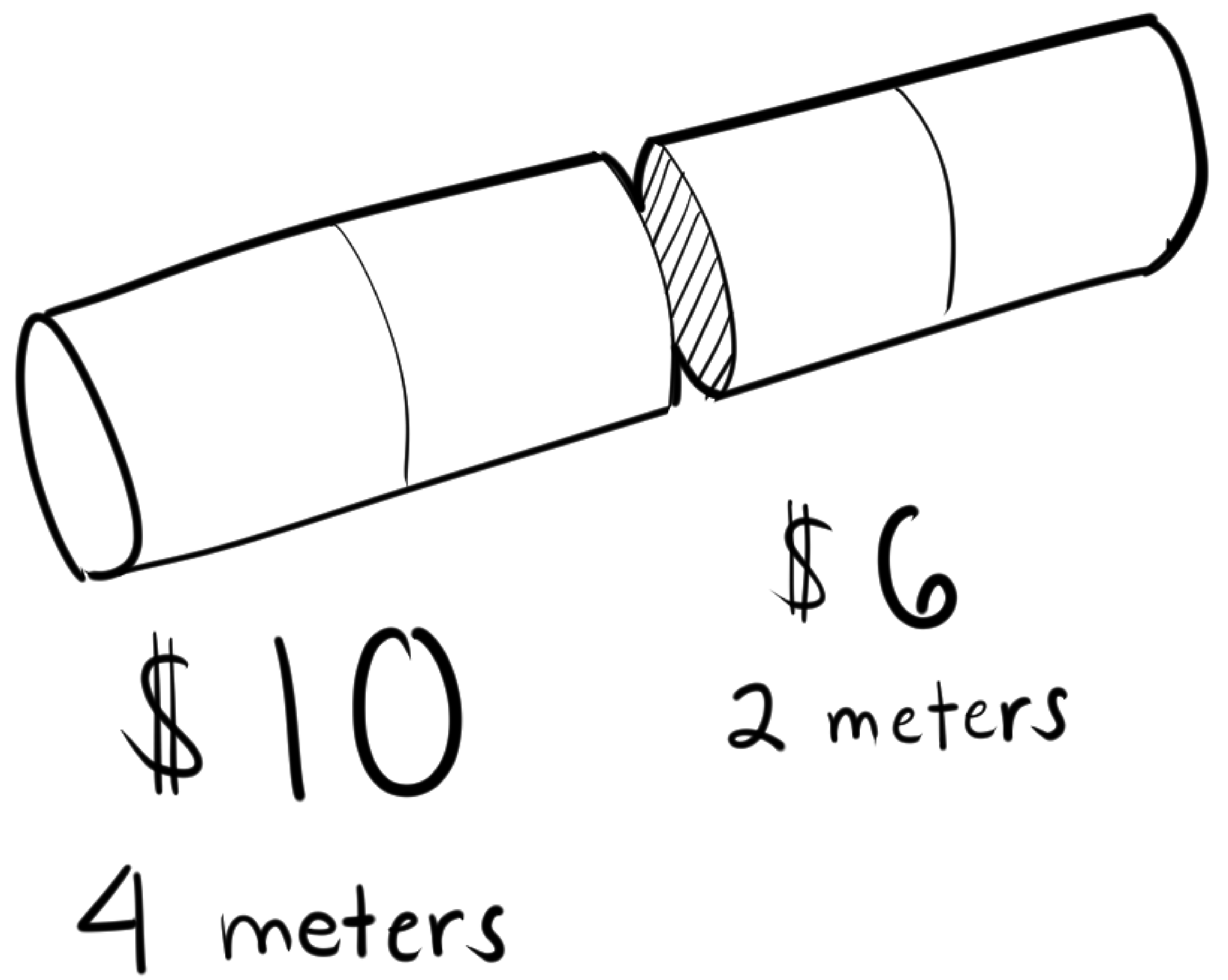


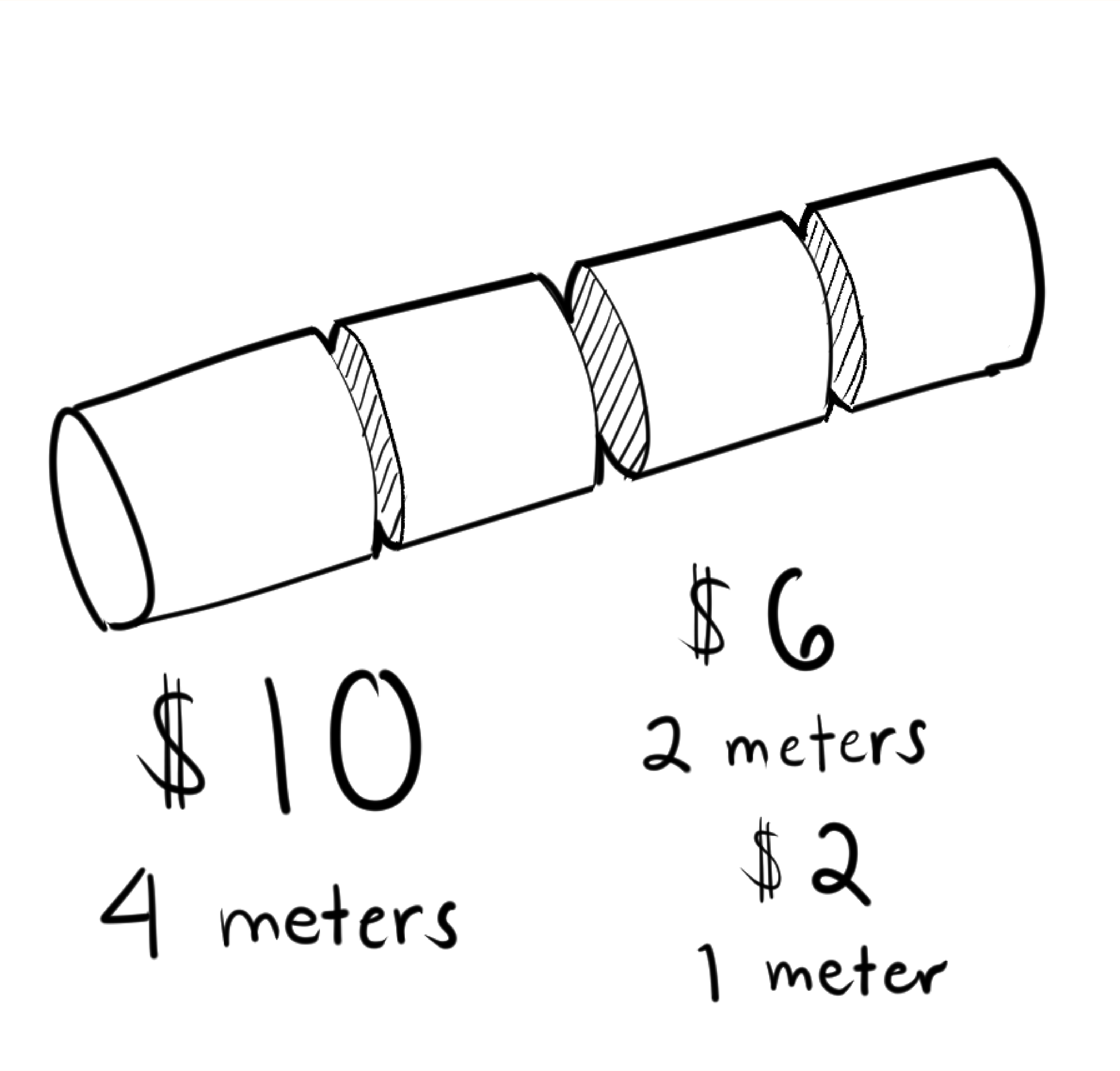
ROD CUTTING

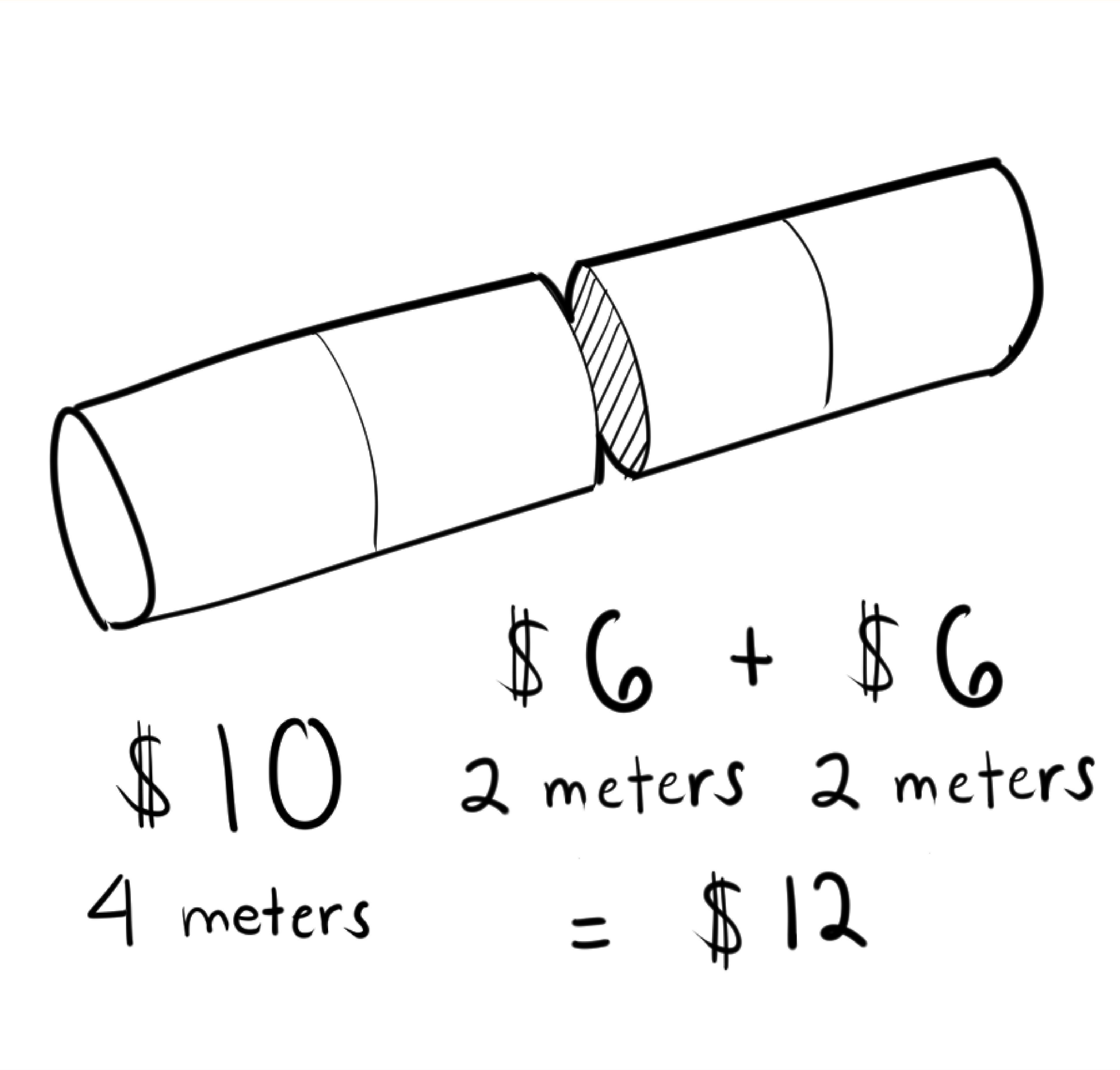


$10

4 meters


$ 10
4 meters
$ 6
2 meters

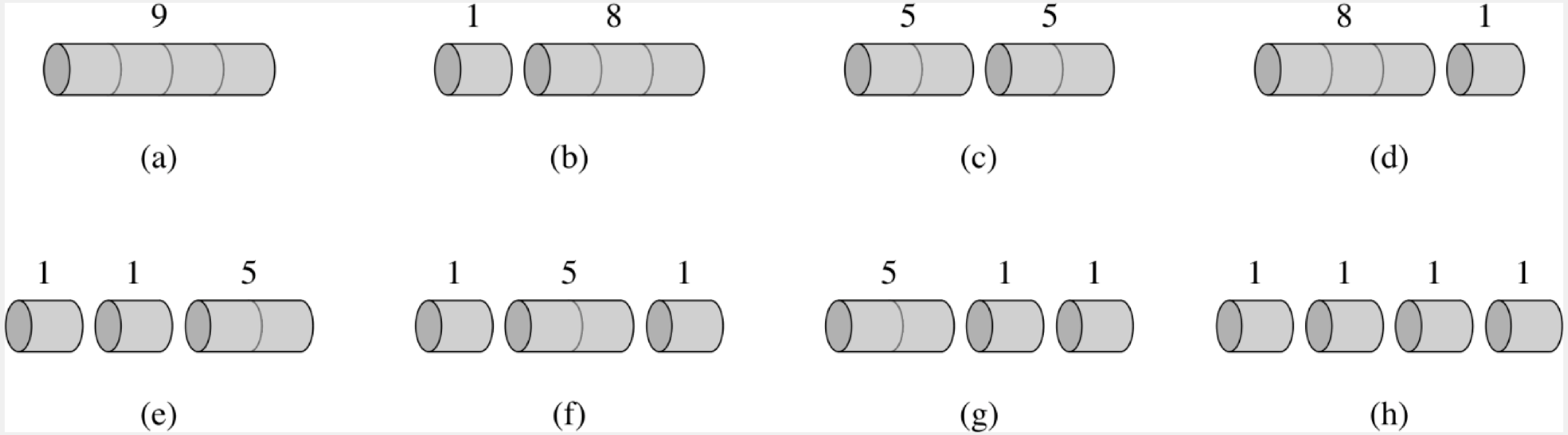

$10
4 meters
$6
2 meters
$2
1 meter


$10
4 meters
$6 + $6
2 meters 2 meters
= $12

ROD CUTTING

Given a rod of length i inches and an array of prices that includes prices of all pieces of size smaller than i. Determine the maximum value obtainable by cutting up the rod and selling the pieces.

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30



SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

we solve for everything from 1 to i

i is the max length
k is the iterating variable or current length we're dealing with
p(k) gets price at position/length k

base case: optimal(1) = 1

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(4-1)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(4-1)
p(2) + optimal(4-2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(4-1)
p(2) + optimal(4-2)
p(3) + optimal(4-3)
p(4) + optimal(4-2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(4-1) $\longrightarrow$ 1 + optimal(3)

p(2) + optimal(4-2)

p(3) + optimal(4-3)

p(4) + optimal(4-2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(4-1) $\longrightarrow$ 1 + optimal(3) = p(1) + optimal(3-1)

p(2) + optimal(4-2)

p(3) + optimal(4-3)

p(4) + optimal(4-2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(4-1) $\longrightarrow$ 1 + optimal(3) = p(1) + optimal(3-1)

p(2) + optimal(4-2)

p(3) + optimal(4-3)

p(4) + optimal(4-2)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) $\longrightarrow$ 1 + optimal(2) = p(1) + optimal(2-1)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

p(2) + optimal(2-2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) → 1 + optimal(2) = p(1) + optimal(2-1)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

p(2) + optimal(2-2)

1 + optimal(2) = p(1) + optimal(1)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) → 1 + optimal(2) = p(1) + optimal(2-1)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

p(2) + optimal(2-2)

1 + optimal(2) = p(1) + optimal(1)

base case: optimal(1) = 1

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) → 1 + optimal(2) = p(1) + optimal(2-1)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

p(2) + optimal(2-2)

1 + optimal(2) = 1 + 1 = 2

base case: optimal(1) = 1

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) $\longrightarrow$ 1 + optimal(2) = 2

p(2) + optimal(3-2)

p(3) + optimal(3-3)

p(2) + optimal(2-2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) ⟶ 1 + optimal(2) = 2

p(2) + optimal(2-2)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) ⟶ 1 + optimal(2) = 2

p(2) + optimal(0)

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) ⟶ 1 + optimal(2) = 2

p(2) + optimal(3-2)

p(3) + optimal(3-3)

p(2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) $\longrightarrow$ 1 + optimal(2) = 2

5

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) $\longrightarrow$ 1 + 5

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = p(1) + optimal(3-1) $\longrightarrow$ 6

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = 6

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = 6

p(2) + optimal(3-2)

p(3) + optimal(3-3)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = 6

5 + optimal(1)

8 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = 6

5 + 1

8

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = 6

6

8

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

1 + optimal(3) = 8

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(4-1)
p(2) + optimal(4-2)
p(3) + optimal(4-3)
p(4) + optimal(4-2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = p(1) + optimal(3)
p(2) + optimal(2)
p(3) + optimal(1)
p(4) + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = 1 + 8
5 + 5
8 + 1
9 + 0

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = 9
10
9
9

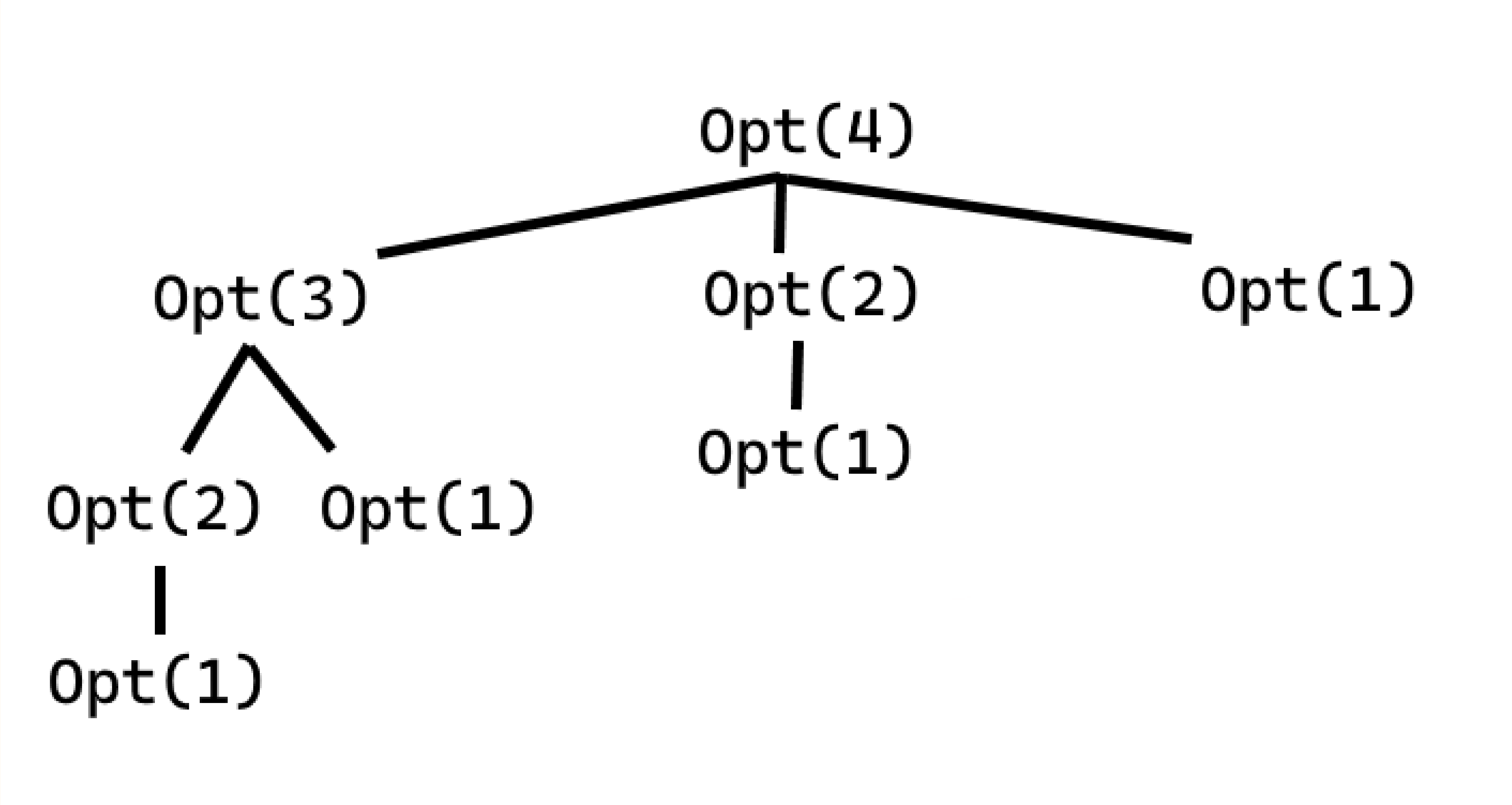
SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

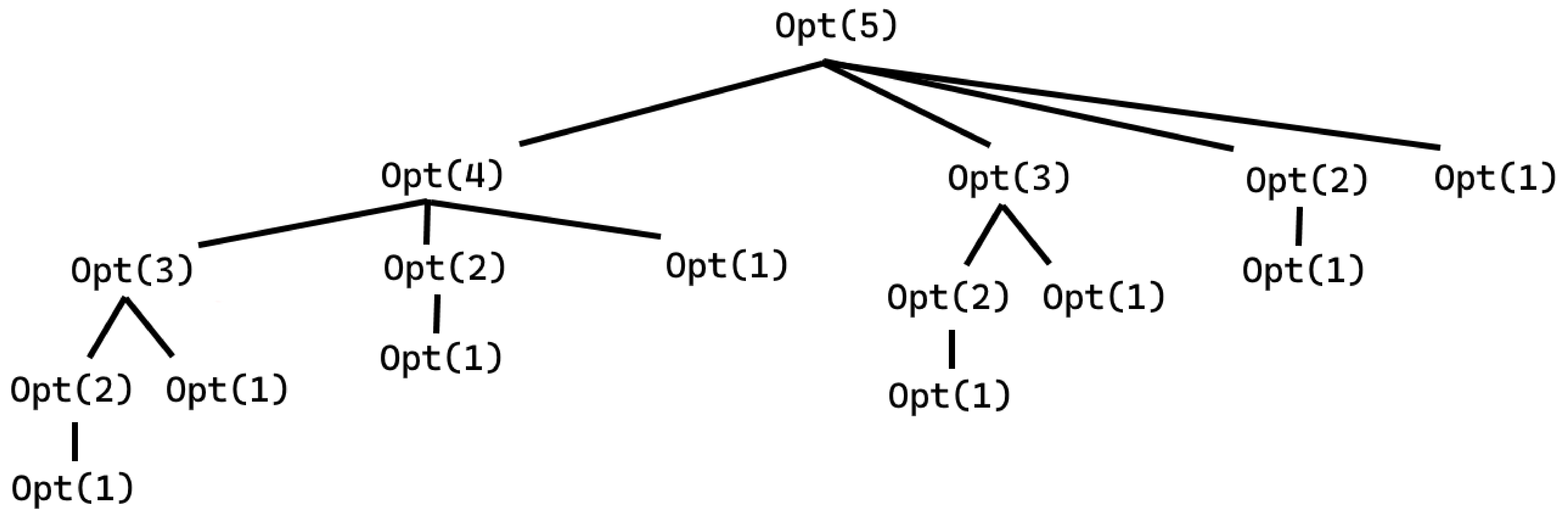
length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimal(4) = 10

SIMULATION



SIMULATION


Opt(5)
Opt(4)
Opt(3)
Opt(2)
Opt(1)
Opt(3)
Opt(2)
Opt(1)
Opt(2)
Opt(1)
Opt(1)
Opt(2)
Opt(1)
Opt(1)
Opt(2)
Opt(1)
Opt(1)
Opt(1)
Opt(1)
Opt(1)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable										

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable										

optimal(5) = p(1) + optimal(5-1)
p(2) + optimal(5-2)
p(3) + optimal(5-3)
p(4) + optimal(5-4)
p(5) + optimal(5-5)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable										

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable										

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable										

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable										

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

optimal(2) = 1 + optimal(1)
5 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable										

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

optimal(2) = 1 + optimal(1)
5 + optimal(0)

optimal(1) = 1

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1									

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

optimal(2) = 1 + optimal(1)
5 + optimal(0)

optimal(1) = 1

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1									

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

optimal(2) = 1 + 1
5

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1									

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

optimal(2) = 2
5

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1									

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

optimal(2) = 5

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

optimalTable

1	2	3	4	5	6	7	8	9	10
1	5								

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + optimal(2)
5 + optimal(1)
8 + optimal(0)

optimal(2) = 5

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5								

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 1 + 5
5 + 1
8

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5								

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 8

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8							

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

optimal(3) = 8

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8							

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + optimal(3)
5 + optimal(2)
8 + optimal(1)
9 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8							

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 1 + 8
5 + 5
8 + 1
9

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8							

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 10

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

optimal(4) = 10

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 1 + optimal(4)
5 + optimal(3)
8 + optimal(2)
9 + optimal(1)
10 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 1 + 10
5 + 8
8 + 5
9 + 1
10

SIMULATION

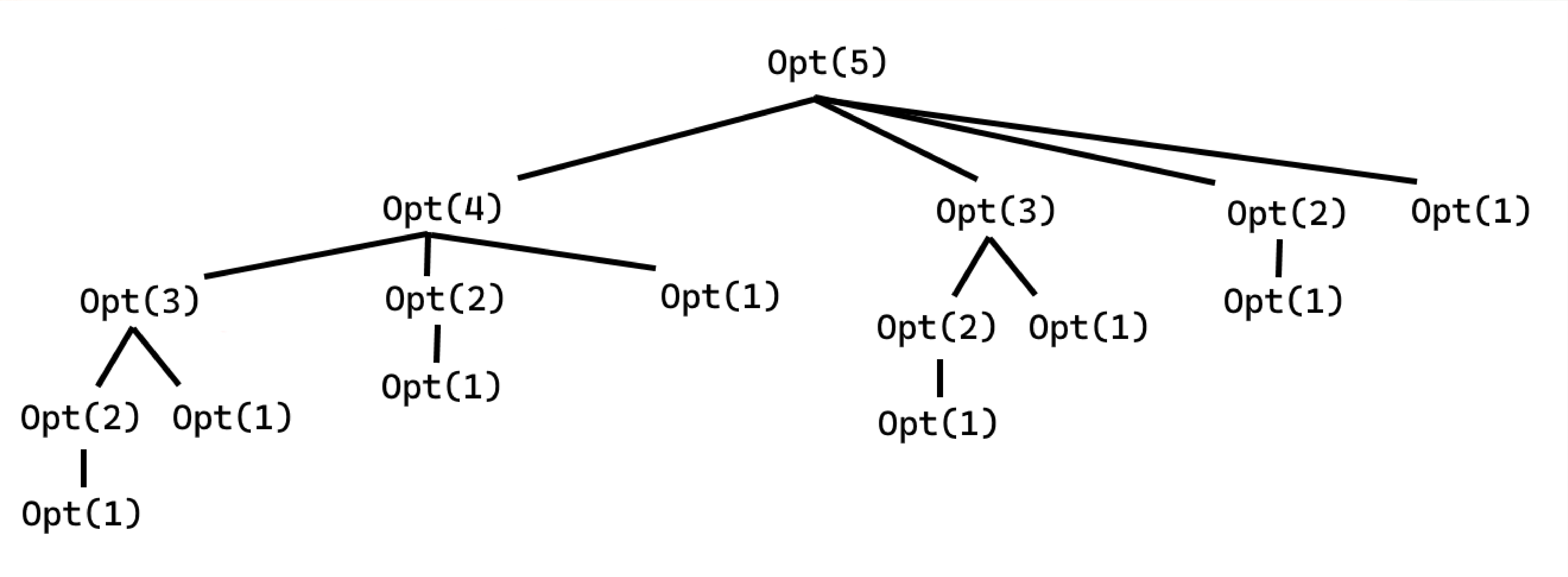
FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

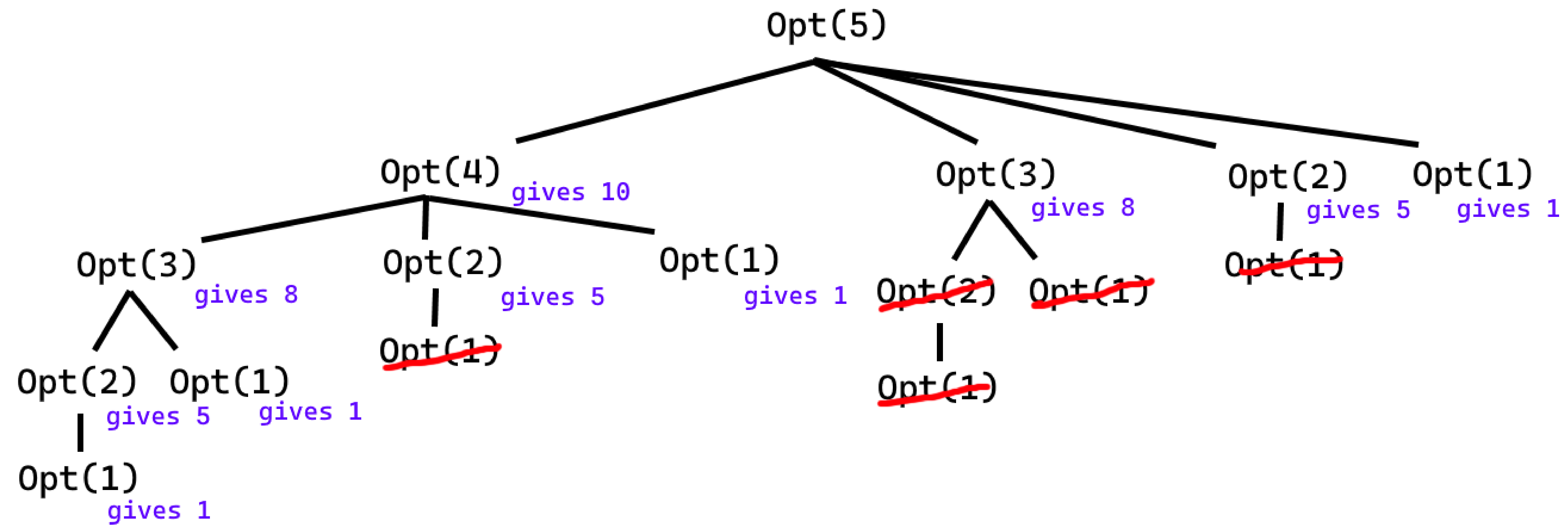
	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 13

SIMULATION


Opt(5)
Opt(4)
Opt(3)
Opt(2)
Opt(1)
Opt(3)
Opt(2)
Opt(1)
Opt(2)
Opt(1)
Opt(1)
Opt(2)
Opt(1)
Opt(1)
Opt(1)
Opt(2)
Opt(1)
Opt(1)
Opt(1)

SIMULATION



Opt(5)
Opt(4)
gives 10
Opt(3)
gives 8
Opt(2)
gives 5
Opt(1)
gives 1
Opt(3)
gives 8
Opt(2)
gives 5
Opt(1)
gives 1
Opt(2)
gives 5
Opt(1)
gives 1
Opt(1)
gives 1
Opt(1)
Opt(2)
Opt(1)
Opt(1)
Opt(1)
gives 5
gives 1
Opt(1)
gives 1

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 5 + optimal(3)
8 + optimal(2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 5 + optimal(3)
8 + optimal(2)

optimal(3) = 8 + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 5 + optimal(3)
8 + optimal(2)

optimal(3) = p(3) + optimal(0)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 5 + optimal(3)
8 + optimal(2)

optimal(3) = p(3) + optimal(0)

↑

length of 3 + optimal length of 0

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 5 + optimal(3)

8 + optimal(2)

length of 2 + optimal length of 3

optimal(3) = p(3) + optimal(0)

length of 3 + optimal length of 0

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 5 + optimal(3)

8 + optimal(2)

optimal(3) = length of 3

length of 2 + optimal length of 3

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = 5 + optimal(3)

8 + optimal(2)

length of 2 + length of 3

optimal(3) = length of 3

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = length of 2 + length of 3

8 + optimal(2)

SIMULATION

FORMULA: optimal(i) = p(k) + optimal(i-k)

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

	1	2	3	4	5	6	7	8	9	10
optimalTable	1	5	8	10						

optimal(5) = length of 2 + length of 3
length of 3 + length of 2

INTERNET CODE VS STREAMLINED CODE

INTERNET CODE

STREAMLINED CODE

```
INT MAX(INT A, INT B) {
    RETURN (A > B) ? A : B;
}
```

INTERNET CODE VS STREAMLINED CODE

INTERNET CODE

```
#define INT_MIN -99999;


int cutRod(int price[], int n){
    if(n<=0){
        return 0;
    }

    int max_val = int_min;

    for(int i=0 ; i<n ; i++){
            max_val = max(max_val, price[i]+cutRod(price,n-i-1));
    }

    return max_val;
}
```

INTERNET CODE VS STREAMLINED CODE

STREAMLINED CODE

```
#define MAX_SIZE 7

int* rodCutting(int price[]){
 int length, newPrice, v, max;
 int* optimal;

 optimal = (int *)malloc(sizeof(int)*MAX_SIZE));

 for(length=0 ; length<MAX_SIZE ; length++){
    max = -99999;
    for(v=0 ; v<length+1 ; v++){
       if(v<length){
          newPrice = price[v] + price[length-v-1];
       }else{
          newPrice = price[v];
       }

       if(max < newPrice){
          max = newPrice;
       }
    }

    optimal[length] = max;
 }

return optimal;
}
```

TIME AND SPACE COMPLEXITY

THE TIME COMPLEXITY OF THE ROD CUTTING PROBLEM IS O(N^2)

THE SPACE COMPLEXITY IS O(N)