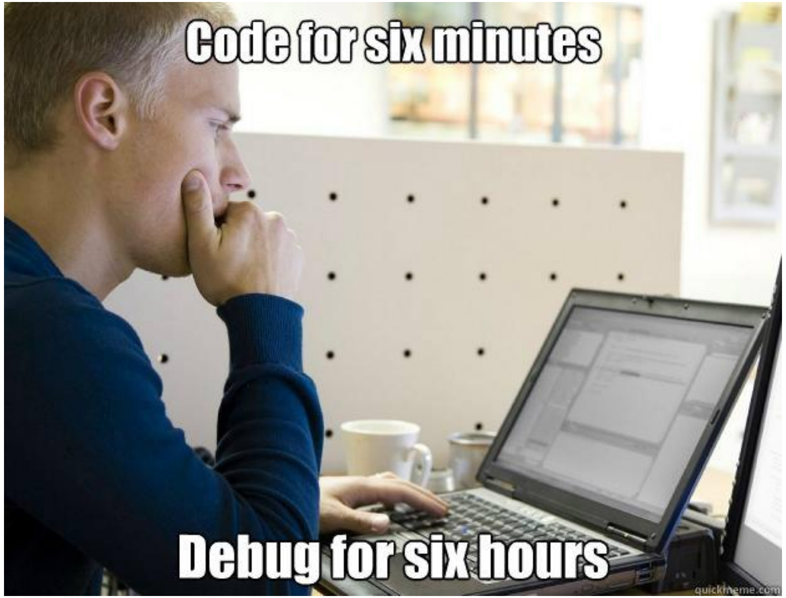
Clean Code

Boston University CS 506 - Lance Galletti

The Problem


Code for six minutes
Debug for six hours
quickmeme.com

99 little bugs in the code.
99 little bugs in the code.
Take one down, patch it around.
127 little bugs in the code...

The Problem

“Software Systems get replaced not when they wear out but when they crumble under their own weight because they have become too complex”

Why Does this Occur?

vocabulary + grammar != poetry

Set yourself up for success

- Stay organized - have a plan and prioritize!
- Structure
    - Do one thing
    - Compartmentalize
    - Many functions with small bodies > one function with a large body
    - Before writing code ask “How will someone use this (or part of this) code?”. Minimize side effects
- Method / Process
    - Top Down Approach
    - Bottom up Approach
    - Solve the 90% problem first - then improve / refine to get to 100%
    - Boring code is good code: keep it simple
    - Check soundness by reading your code before testing

Clean Code

'Clean code' by Robert C. Martin

Github Gist Summary of the Book:
https://gist.github.com/wojteklu/73c6914cc446146b8b533c0988cf8d29

But if you must debug

1. **Don’t Panic!**
2. Read the error
    - What is the error telling you?
    - Where did the error occur?
    - Is this a cause or just a symptom (one bug can hide another)?
3. Re-read your code - take your time!
    - Can you mentally trace through your code to reproduce the error in your head?
        - If not: your code may need some refactoring because it’s too complex!
4. Sanity check where you can
    - Is everything set up properly? Are the things that are supposed to communicate actually communicating?
5. Now look online for some help
    - Hopefully with the above out of the way you have a good idea what to search for in order to actually fix the issue
6. Take breaks

Example: Max Path Sum

Starting at the top of the triangle and moving down to adjacent numbers below: Find the path from the root to a leaf with the maximum sum.

```
   3
  7 4
 2 4 6
8 5 9 3
```

Example: Max Path Sum

Starting at the top of the triangle and moving down to adjacent numbers below: Find the path from the root to a leaf with the maximum sum.

3

7 4

2 4 6

8 5 9 3

Example: Max Path Sum

Starting at the top of the triangle and moving down to adjacent numbers below: Find the path from the root to a leaf with the maximum sum.

3
7 4
2 4 6
8 5 9 3

Example: Max Path Sum

Starting at the top of the triangle and moving down to adjacent numbers below: Find the path from the root to a leaf with the maximum sum.

```
      3
     7 4
    2 4 6
   8 5 9 3
```

Example: Max Path Sum

Starting at the top of the triangle and moving down to adjacent numbers below: Find the path from the root to a leaf with the maximum sum.

```
   3
  7 4
 2 4 6
8 5 9 3
```

Example: Max Path Sum

Starting at the top of the triangle and moving down to adjacent numbers below: Find the path from the root to a leaf with the maximum sum.

```
      3
    7   4
  2   4   6
8   5   9   3
```

In the above example the max path sum in 3 + 7 + 4 + 9 = 23

Example: Max Path Sum

Starting at the top of the triangle and moving down to adjacent numbers below: Find the path from the root to a leaf with the maximum sum.

```
   3
  7 4
 2 4 6
8 5 9 3
```

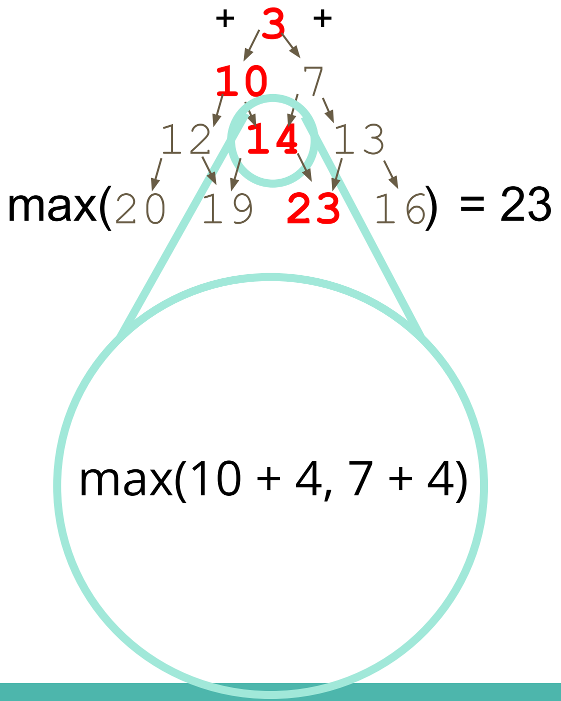
In the above example the max path sum in 3 + 7 + 4 + 9 = 23

Brute Force: Find all paths and get the max.

Source: https://projecteuler.net/problem=18

Example: Max Path Sum

Our Algorithm



Coding steps

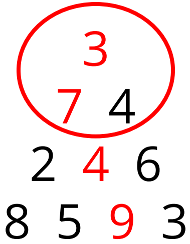
3

7 4

2 **4** 6

8 5 **9** 3

Coding steps



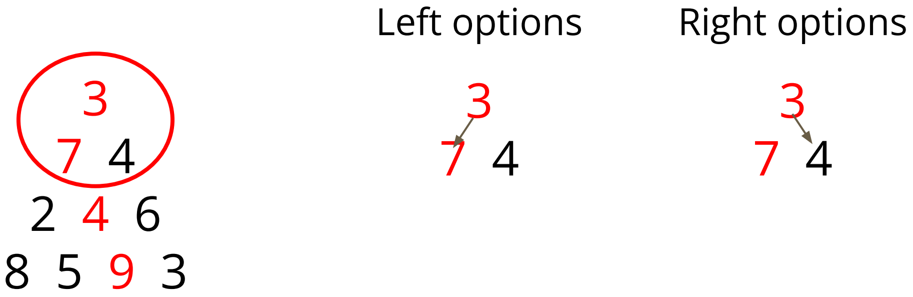
3

7 4

2 4 6

8 5 9 3

Coding steps



3
7 4
2 4 6
8 5 9 3

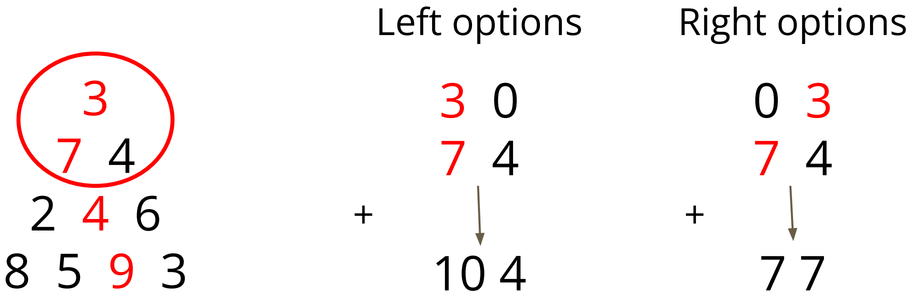
Left options

3
7 4

Right options

3
7 4

Coding steps



Left options

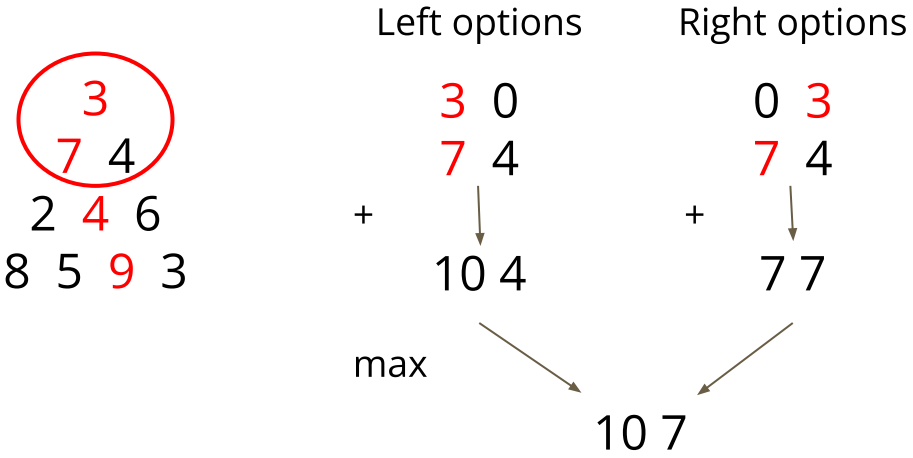
Right options

3
7 4
2 4 6
8 5 9 3

3 0
7 4
+
10 4

0 3
7 4
+
7 7

Coding steps



3
7 4
2 4 6
8 5 9 3

Left options

3 0
7 4
+
10 4

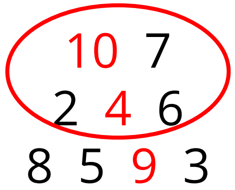
Right options

0 3
7 4
+
7 7

max

10 7

Coding steps

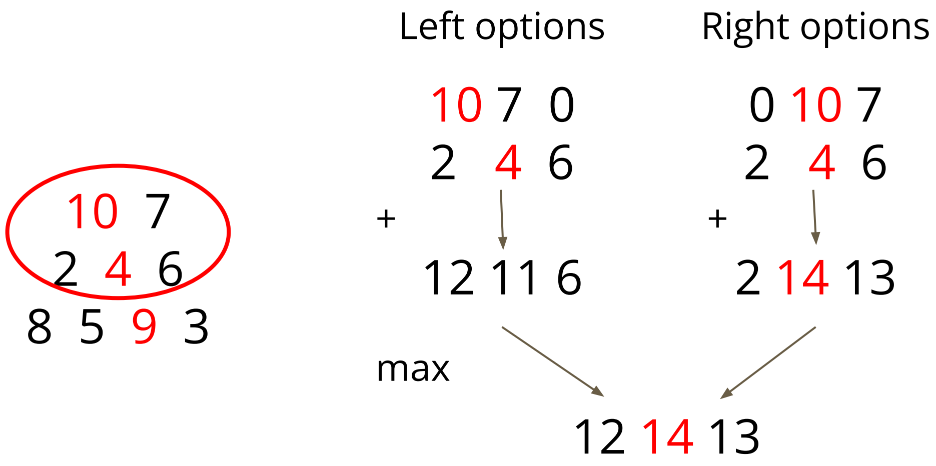


10 7

2 4 6

8 5 9 3

Coding steps



10 7

2 4 6

8 5 9 3

Left options

10 7 0

2 4 6

+

12 11 6

Right options

0 10 7

2 4 6

+

2 14 13

max

12 14 13