Not All Instances of Hard Problems are Difficult[†]

Will Bryan and Andrew Mertz

February 23, 2024

[†]and they can be a lot of fun

Overview

In this talk, we explore some problems from Advent of Code 2023 and the techniques that make these problems simpler than they first appear.

We will also look at some problems just for fun.

What is AoC?

Advent of Code is an annual series of small programming puzzles for a variety of skill sets and skill levels in any programming language you like.

It runs from December 1^{st} to December 25^{th} each year (since 2015).

Here are the current completion statistics for each day. Gold indicates users that have completed both parts of a puzzle, while silver indicates users that have completed only the first half. Each * or * star represents up to 7755 users.

```
25   11188   3070  ***
24   12504   5014  ***
23   15233   2992  ***
22   16484   1001  ****
21   14936  10316  ****
20   18964   4486  ****
19   23210   7295  ****
18   25496   4988  *****
17   24701   1128  *****
16   36168   1061  ******
15   41723   4250  *******
14   37274   7582  ******
13   38957   5309  *******
12   31750  14617  *******
11   58390   2438  *********
10   49160  17346  **********
 9   77739   1283  ***********
 8   75250  14948  ***********
 7   83178   7468  ***********
 6  104874   1985  ***************
 5   81386  31578  ****************
 4  132136  18012  ********************
 3  132321  20318  ********************
 2  199469   9535  ****************************
 1  234418  75774  ****************************************
```

adventofcode.com/2023/stats

Private Leaderboard

```
This is the private leaderboard of Will for Advent of Code 2023. You can
use a different [Ordering], manage your [Private Leaderboards], use an
[API], or switch to another [Event].

Gold indicates the user got both stars for that day, silver means just the
first star, and gray means none.

                   1111111111222222
         12345678901234567890123456
 1)  478 **************************  ZeroTau (AoC++)
 2)  420 *************************.  Will (AoC++)
 3)  352 ******************..**..**  Sgr A8
 4)  182 ************............  one-reader
 5)   53 *****.....................  Austin Stortzum
 6)   37 ****......................  zemkat
 7)   25 ***.......................  Jack Schmidt
 8)    7 *........................  Abby Gamboa
 9)    2 *........................  Guyfee
10)    0 ..........................  (anonymous user #3280719)
```

Global Leaderboard

Below is the Advent of Code 2023 overall leaderboard; these are the 100 users with the highest total score. Getting a star first is worth 100 points, second is 99, and so on down to 1 point at 100th place.

You can change how you appear here on the [Settings] page. You can also view your own [Personal Times] or use a [Private Leaderboard].

```
  1) 3257   xiaowuc1
  2) 3174   tckmn
  3) 2909   5space (AoC++)
  4) 2486   nthistle (AoC++) (Sponsor)
  5) 2484   jonathanpaulson (AoC++)
  6) 2476   Antonio Molina (AoC++) (Sponsor)
  7) 2404   dan-simon
  8) 2370   bluepichu
  9) 2285   leijurv (AoC++)
 10) 2241   boboquack
 11) 2226   hyper-neutrino
 12) 2198   D. Salgado
 13) 2092   Ian DeHaan
 14) 2046   Noble Mushtak
 15) 2034   Anish Singhani (AoC++)
```

adventofcode.com/2023/leaderboard

Why do contests?

- Fun
- Learning
- Community
- Profit?

Once you see it...

Day 1: Sum of Digits

This problem asks us to parse lines of input to find the first and last digits contained within.

Then combine the first digit and the last digit to form a single two-digit number, and sum all such numbers.

The catch is the digits could be spelled out or written as numbers.

Day 1: Example

```
two1nine
eightwothree
abcone2threexyz
xtwone3four
4nineeightseven2
zoneight234
7pqrstsixteen
```

Yields the sum $29 + 83 + 13 + 24 + 42 + 14 + 76 = 281$.

Day 1: Possible Approaches

- Use a regular expression
- Build our own parser
- Use tools like sed or awk
- Use a parser generator like ANTLR

Note the input is small (around 22KB).

While we can find the digits with only one pass over the input. Even if we take multiple passes, we can still solve the problem quickly for input this small.

Day 1: Possible Approaches

- Use a regular expression
- Build our own parser
- Use tools like sed or awk
- Use a parser generator like ANTLR

Note the input is small (around 22KB).

While we can find the digits with only one pass over the input. Even if we take multiple passes, we can still solve the problem quickly for input this small.

Day 1: Solutions

- Python
- Bash pipeline
- Circuit

Questions?

This talk available at `github.com/ZeroTau/AoC2023Talk`

Thank You!