Advent of Code Day 1, 2020

Joel Laxamana

2023-04-04

— Day 1: Report Repair — After saving Christmas five years in a row, you've decided to take a vacation at a nice resort on a tropical island. Surely, Christmas will go on without you.

The tropical island has its own currency and is entirely cash-only. The gold coins used there have a little picture of a starfish; the locals just call them stars. None of the currency exchanges seem to have heard of them, but somehow, you'll need to find fifty of these coins by the time you arrive so you can pay the deposit on your room.

To save your vacation, you need to get all fifty stars by December 25th.

Collect stars by solving puzzles. Two puzzles will be made available on each day in the Advent calendar; the second puzzle is unlocked when you complete the first. Each puzzle grants one star. Good luck!

Before you leave, the Elves in accounting just need you to fix your expense report (your puzzle input); apparently, something isn't quite adding up.

Specifically, they need you to find the two entries that sum to 2020 and then multiply those two numbers together.

For example, suppose your expense report contained the following:

1721 979 366 299 675 1456

In this list, the two entries that sum to 2020 are 1721 and 299. Multiplying them together produces 1721 * 299 = 514579, so the correct answer is 514579.

Of course, your expense report is much larger. Find the two entries that sum to 2020; what do you get if you multiply them together?

```
# load packages
library("plyr")
library("tidyr")
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# # read in data
my_data <- read.table(file.path("/cloud/project/raw", "aoc_2020_1_input.txt"), blank.lines.skip = F) %>%
  rename(expense=V1)
```

```r
# count of expense
n <- nrow(my_data)
n
```

```
## [1] 200
```

```r
# create all possible combinations of 2 to get sum
comb <- as.data.frame(combn(my_data$expense, 2))
comb %>% select(V1:V10)
```

```
##     V1   V2   V3   V4   V5   V6   V7   V8   V9  V10
## 1 1780 1780 1780 1780 1780 1780 1780 1780 1780 1780
## 2 1693 1830 1756 1858 1868 1968 1809 1996 1962 1800
```

```r
# transpose all possible combinations into 2 columns
tcomb <- as_tibble(t(comb))
```

```
## Warning: The `x` argument of `as_tibble.matrix()` must have unique column names if
## `.name_repair` is omitted as of tibble 2.0.0.
## i Using compatibility `.name_repair`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
tcomb
```

```
## # A tibble: 19,900 x 2
##       V1    V2
##    <int> <int>
##  1  1780  1693
##  2  1780  1830
##  3  1780  1756
##  4  1780  1858
##  5  1780  1868
##  6  1780  1968
##  7  1780  1809
##  8  1780  1996
##  9  1780  1962
## 10  1780  1800
## # i 19,890 more rows
```

```r
# sum the columns
tcombsum <- tcomb %>% mutate(sum = V1+V2)
tcombsum
```

```
## # A tibble: 19,900 x 3
##       V1    V2   sum
##    <int> <int> <int>
##  1  1780  1693  3473
##  2  1780  1830  3610
##  3  1780  1756  3536
##  4  1780  1858  3638
##  5  1780  1868  3648
##  6  1780  1968  3748
```

```
##  7  1780  1809  3589
##  8  1780  1996  3776
##  9  1780  1962  3742
## 10  1780  1800  3580
## # i 19,890 more rows
```

```
# find sum that equals the number you want
want <- tcombsum %>% filter(sum==2020)
want
```

```
## # A tibble: 1 x 3
##      V1    V2   sum
##   <int> <int> <int>
## 1  1078   942  2020
```

```
# multiply those 2 numbers
fix <- want %>% mutate(product = V1*V2)
```

PART 1 : Solution

```
fix$product
```

```
## [1] 1015476
```

— Part Two — The Elves in accounting are thankful for your help; one of them even offers you a starfish coin they had left over from a past vacation. They offer you a second one if you can find three numbers in your expense report that meet the same criteria.

Using the above example again, the three entries that sum to 2020 are 979, 366, and 675. Multiplying them together produces the answer, 241861950.

In your expense report, what is the product of the three entries that sum to 2020?

```
# create all possible combinations of 3 to get sum
comb3 <- as.data.frame(combn(my_data$expense, 3))
```

```
# transpose all possible combinations into 3 columns
tcomb3 <- as_tibble(t(comb3))
head(tcomb3)
```

```
## # A tibble: 6 x 3
##      V1    V2    V3
##   <int> <int> <int>
## 1  1780  1693  1830
## 2  1780  1693  1756
## 3  1780  1693  1858
## 4  1780  1693  1868
## 5  1780  1693  1968
## 6  1780  1693  1809
```

```
# sum the columns
tcombsum3 <- tcomb3 %>% mutate(sum = V1+V2+V3)
head(tcombsum3)
```

```
## # A tibble: 6 x 4
##      V1    V2    V3   sum
##   <int> <int> <int> <int>
## 1  1780  1693  1830  5303
## 2  1780  1693  1756  5229
```

```
## 3  1780  1693  1858  5331
## 4  1780  1693  1868  5341
## 5  1780  1693  1968  5441
## 6  1780  1693  1809  5282
```

```
# find sum that equals the number you want
want3 <- tcombsum3 %>% filter(sum==2020)
want3
```

```
## # A tibble: 1 x 4
##      V1    V2    V3   sum
##   <int> <int> <int> <int>
## 1   956   262   802  2020
```

```
# multiply those 3 numbers
fix3 <- want3 %>% mutate(product = V1*V2*V3)
```

PART 2 : Solution

```
fix3$product
```

```
## [1] 200878544
```