Advent of Code 2021 - Day 3 Speed Edition

Gus Lipkin ~ github.com/guslipkin/AdventOfCode2021

Fastest solution

```
dtM <- data.frame("V1" = readLines("input.txt"))
dtM <- tidyr::separate(dtM, "V1", c("d", "n"), sep = " ", convert = TRUE)
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 1000 rows [1, 2,
## 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
# part 1
dt <- dtM
dt$n <- ifelse(dt$d == "down", dt$n * -1, dt$n)
abs(sum(dt$n[dt$d == "forward"]) * sum(dt$n[dt$d != "forward"]))
```

```
## [1] NA
```

```
# part 2
dt <- dtM
dt$aim <- cumsum(ifelse(dt$d == "up", dt$n * -1,
                        ifelse(dt$d == "forward", 0, dt$n)))
sum(ifelse(dt$d == "forward", dt$n, 0), na.rm = TRUE) *
  sum(ifelse(dt$d == "forward", dt$aim * dt$n, 0), na.rm = TRUE)
```

```
## [1] 0
```

Benchmark

```
rbenchmark::benchmark(
  "First try" = {
    library(tidyverse)
    library(data.table)
    library(compositions)

    # part 1
    dt <- data.frame(fread("input.txt", sep = "\n",
                           colClasses = c("character")))
    dt <- tidyr::separate(dt, "V1", paste0("V", 1:13),
                          sep = "", convert = TRUE)
```

```
get_mode <- function(x)
  return(names(sort(table(x), decreasing = T, na.last = T)[1]))
o <- c()
for (i in 2:13)
  o <- append(o, get_mode(dt[, i]))
o <- unbinary(paste(o, collapse = ""))

get_mode <- function(x)
  return(names(sort(table(x), na.last = T)[1]))
c <- c()
for (i in 2:13)
  c <- append(c, get_mode(dt[, i]))
c <- unbinary(paste(c, collapse = ""))

o * c

# part 2
dtM <- data.frame(fread("input.txt", sep = "\n",
                        colClasses = c("character")))
dtM <- tidyr::separate(dtM, "V1", paste0("V", 1:13),
                       sep = "", convert = TRUE)
dtM <- dtM %>% select(-"V1")
dt <- dtM

get_modeMax <- function(x) {
  temp <- sort(table(x), decreasing = T, na.last = T)
  ifelse(temp[1] == temp[2], return(1), names(temp[1]))
}
o <- c()
for (i in 1:12) {
  dt <- dt %>% filter_at(i, all_vars(. == get_modeMax(dt[, i])))
  if (nrow(dt) == 1) {
    oxygen <- paste(as.character(dt[1, ]), collapse = "")
    break
  }
}

dt <- dtM
get_modeMin <- function(x) {
  temp <- sort(table(x), na.last = T)
  ifelse(temp[1] == temp[2], return(0), names(temp[1]))
}
c <- c()
for (i in 1:12) {
  dt <- dt %>% filter_at(i, all_vars(. == get_modeMin(dt[, i])))
  if (nrow(dt) == 1) {
    co2 <- paste(as.character(dt[1, ]), collapse = "")
    break
  }
}

unbinary(oxygen) * unbinary(co2)
},
```

```
"Second try" = {
library(tidyverse)
library(data.table)
library(compositions)

# part 1
dt <- fread("input.txt", sep = "\n", colClasses = c("character")) %>%
  as.data.frame() %>%
  tidyr::separate("V1", paste0("V", 1:13), sep = "", convert = TRUE) %>%
  select(-"V1")

getModeMin <- function(x)
  return(names(sort(table(x), na.last = T)[1]))
getModeMax <- function(x)
  return(names(sort(table(x), decreasing = T, na.last = T)[1]))

o <- unbinary(paste(apply(dt, 2, getModeMax), collapse = ""))

c <- unbinary(paste(apply(dt, 2, getModeMin), collapse = ""))

o * c

# # part 2
dtM <- fread("input.txt", sep = "\n", colClasses = c("character")) %>%
  as.data.frame() %>%
  tidyr::separate("V1", paste0("V", 1:13), sep = "", convert = TRUE) %>%
  select(-"V1")
dto <- dtM
dtc <- dtM

getModeMin <- function(x) {
  temp <- sort(table(x), na.last = T)
  ifelse(temp[1] == temp[2], return(0), names(temp[1]))
}
getModeMax <- function(x) {
  temp <- sort(table(x), decreasing = T, na.last = T)
  ifelse(temp[1] == temp[2], return(1), names(temp[1]))
}

for (i in 1:12) {
  if(nrow(dto) > 1)
    dto <- dto %>% filter_at(i, all_vars(. == getModeMax(dto[, i])))
  if(nrow(dtc) > 1)
    dtc <- dtc %>% filter_at(i, all_vars(. == getModeMin(dtc[, i])))
  if (nrow(dto) == 1 & nrow(dtc) == 1) {
    oxygen <- paste(as.character(dto[1, ]), collapse = "")
    co2 <- paste(as.character(dtc[1, ]), collapse = "")
    break
  }
}

unbinary(oxygen) * unbinary(co2)
},
```

```
  replications = 10, columns = c(1:5), order = "user.self")
```

```
##         test replications user.self sys.self elapsed
## 2 Second try           10     0.678    0.017   0.696
## 1  First try           10     0.683    0.008   0.691
```