Advent of Code 2021 - Day 6 Speed Edition

Gus Lipkin ~ github.com/guslipkin/AdventOfCode2021

This was a classic Advent Of Code misdirection. Once I figured it out, my solution was quick to write and execute.

Fastest solution

```
# part 1 and 2
dt2 <- as.numeric(data.table::fread("input.txt", header = FALSE)[1])

t <- table(dt2)
v <- c(0, 0, t, rep(0, 8 - length(t)))

for (i in 1:256) {
  v <- c(v[2:7], v[8] + v[1], v[9:10], v[2])
  if (i == 80)
    print(sum(v))
}
```

```
## [1] 363101
```

```
options(scipen = 999)
sum(v)
```

```
## [1] 1644286074024
```

Benchmark

```
bench <- rbenchmark::benchmark(
  "First try" = {
    # part 1
    dt <- as.numeric(unlist(stringr::str_split(readLines("input.txt"), ",")))
    dt2 <- dt
    for(d in 1:80) {
      dt2 <- dt2 - 1
      if(sum(dt2 == -1) > 0) {
        dt2 <- append(dt2, rep(8, sum(dt2 == -1)))
        dt2[dt2 == -1] <- 6
        }
      }
```

```r
s <- length(dt2)

# part 2
dt <- as.numeric(unlist(stringr::str_split(readLines("input2.txt"), ",")))
dt2 <- dt

dt <- data.frame("m" = 0, "zero" = 0, "one" = 0, "two" = 0,
                 "three" = 0, "four" = 0, "five" = 0, "six" = 0,
                 "seven" = 0, "eight" = 0)

for(i in 1:length(dt2)) {
  if(dt2[i] == 0) {dt$zero[1] <- dt$zero[1] + 1}
  if(dt2[i] == 1) {dt$one[1] <- dt$one[1] + 1}
  if(dt2[i] == 2) {dt$two[1] <- dt$two[1] + 1}
  if(dt2[i] == 3) {dt$three[1] <- dt$three[1] + 1}
  if(dt2[i] == 4) {dt$four[1] <- dt$four[1] + 1}
  if(dt2[i] == 5) {dt$five[1] <- dt$five[1] + 1}
  if(dt2[i] == 6) {dt$six[1] <- dt$six[1] + 1}
  if(dt2[i] == 7) {dt$seven[1] <- dt$seven[1] + 1}
  if(dt2[i] == 8) {dt$eight[1] <- dt$eight[1] + 1}
}

for (i in 1:255) {
  dt$zero[1] <- dt$one[1]
  dt$one[1] <- dt$two[1]
  dt$two[1] <- dt$three[1]
  dt$three[1] <- dt$four[1]
  dt$four[1] <- dt$five[1]
  dt$five[1] <- dt$six[1]
  dt$six[1] <- dt$seven[1] + dt$m[1]
  dt$seven[1] <- dt$eight[1]

  dt$eight[1] <- dt$m[1]
  dt$m[1] <- dt$zero[1]
}

options(scipen = 999)
s <- sum(dt[1, ])
},
"Second try" = {
  # part 1
  dt2 <- as.numeric(data.table::fread("input.txt", header = FALSE)[1])

  t <- table(dt2)
  v <- c(0, 0, t, rep(0, 8 - length(t)))

  for (i in 1:80)
    v <- c(v[2:7], v[8] + v[1], v[9:10], v[2])
  s <- sum(v)

  # part 2
  dt2 <- as.numeric(data.table::fread("input.txt", header = FALSE)[1])
```

```
      t <- table(dt2)
      v <- c(0, 0, t, rep(0, 8 - length(t)))

      for (i in 1:256)
        v <- c(v[2:7], v[8] + v[1], v[9:10], v[2])
      options(scipen = 999)
      s <- sum(v)
    },
  "Third try" = {
      # part 1 and 2
      dt2 <- as.numeric(data.table::fread("input.txt", header = FALSE)[1])

      t <- table(dt2)
      v <- c(0, 0, t, rep(0, 8 - length(t)))

      for (i in 1:256) {
        v <- c(v[2:7], v[8] + v[1], v[9:10], v[2])
        if(i == 80)
          s <- sum(v)
      }
      options(scipen = 999)
      s <- sum(v)
    },
  replications = 1000, columns = c(1:5), order = "user.self")

bench$per <- bench$user.self / bench$replications
bench
```

```
##         test replications user.self sys.self elapsed      per
## 3  Third try         1000     2.613    0.057   2.671 0.002613
## 2 Second try         1000     3.964    0.095   4.061 0.003964
## 1  First try         1000    55.858    7.765  63.638 0.055858
```

> None of these will work without lots and lots of computing resources They are showcased here doing part 1 (80 days) for the example sequence

```
dt <- as.numeric(unlist(stringr::str_split(readLines("input2.txt"), ",")))

bench <- rbenchmark::benchmark(
  "simple loop" = {
    dt2 <- dt
    for (d in 1:80) {
      dt2 <- dt2 - 1
      dt2 <- append(dt2, rep(8, sum(dt2 == -1)))
      dt2[dt2 == -1] <- 6
    }
    s <- length(dt2)
  },
  "recursion" = {
    dt2 <- dt
    fish <- function(x, y) {
      if (y == 0) {
        return(length(x))
      } else {
        x <- x - 1
        x <- append(x, rep(8, sum(x == -1)))
        x[x == -1] <- 6
        return(fish(x, y - 1))
      }
    }
    s <- fish(dt2, 80)
  },
  "nested loop" = {
    dt2 <- dt
    for (d in 1:80) {
      l <- length(dt2)
      for (i in 1:l) {
        if (dt2[i] == 0) {
          dt2[i] <- 6
          dt2 <- append(dt2, 8)
        } else {
          dt2[i] <- dt2[i] - 1
        }
      }
    }
    s <- length(dt2)
  },
  replications = 1000, columns = c(1:5), order = "user.self")

bench$per <- bench$user.self / bench$replications
bench
```

```
##          test replications user.self sys.self elapsed      per
## 2   recursion         1000     0.569    0.099   0.667 0.000569
## 1 simple loop         1000     1.806    0.118   1.924 0.001806
## 3 nested loop         1000    35.563   10.673  46.244 0.035563
```