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

```r
# part 1 and 2
dt2 <- c(1,3,3,4,5,1,1,1,1,1,1,2,1,4,1,1,1,5,2,2,4,3,1,1,2,5,4,2,2,3,1,2,3,2,1,
         1,4,4,2,4,4,1,2,4,3,3,3,1,1,3,4,5,2,5,1,2,5,1,1,1,3,2,3,3,1,4,1,1,4,1,
         4,1,1,1,1,5,4,2,1,2,2,5,5,1,1,1,1,2,1,1,1,1,3,2,3,1,4,3,1,1,3,1,1,1,1,
         3,3,4,5,1,1,5,4,4,4,4,2,5,1,1,2,5,1,3,4,4,1,4,1,5,5,2,4,5,1,1,3,1,3,1,
         4,1,3,1,2,2,1,5,1,5,1,3,1,3,1,4,1,4,5,1,4,5,1,1,5,2,2,4,5,1,3,2,4,2,1,
         1,1,2,1,2,1,3,4,4,2,2,4,2,1,4,1,3,1,3,5,3,1,1,2,2,1,5,2,1,1,1,1,1,5,4,
         3,5,3,3,1,5,5,4,4,2,1,1,1,2,5,3,3,2,1,1,1,5,5,3,1,4,4,2,4,2,1,1,1,5,1,
         2,4,1,3,4,4,2,1,4,2,1,3,4,3,3,2,3,1,5,3,1,1,5,1,2,2,4,4,1,2,3,1,2,1,1,
         2,1,1,1,2,3,5,5,1,2,3,1,3,5,4,2,1,3,3,4)

fishTable <- c(0, 0, table(dt2), rep(0, 8 - max(dt2)))

fish <- function(v, d) {
  if (d == 0) {
    return(sum(v, digits = 999))
  } else if (d == 176) {
    print(sum(v))
  }
  fish(c(v[2:7], v[8] + v[1], v[9:10], v[2]), d - 1)
}
fish(fishTable, 256)
```

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

```
## [1] 1.644286e+12
```

Benchmark

```r
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
      }
    }
  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, ])
  },
"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)
    },
  "With recursion!" = {
    # part 1 and 2
    dt2 <- c(1,3,3,4,5,1,1,1,1,1,1,2,1,4,1,1,1,5,2,2,4,3,1,1,2,5,4,2,2,3,1,2,3,
             2,1,1,4,4,2,4,4,1,2,4,3,3,3,1,1,3,4,5,2,5,1,2,5,1,1,1,3,2,3,3,1,4,
             1,1,4,1,4,1,1,1,1,5,4,2,1,2,2,5,5,1,1,1,1,2,1,1,1,1,3,2,3,1,4,3,1,
             1,3,1,1,1,1,3,3,4,5,1,1,5,4,4,4,4,2,5,1,1,2,5,1,3,4,4,1,4,1,5,5,2,
             4,5,1,1,3,1,3,1,4,1,3,1,2,2,1,5,1,5,1,3,1,3,1,4,1,4,5,1,4,5,1,1,5,
             2,2,4,5,1,3,2,4,2,1,1,1,2,1,2,1,3,4,4,2,2,4,2,1,4,1,3,1,3,5,3,1,1,
             2,2,1,5,2,1,1,1,1,1,5,4,3,5,3,3,1,5,5,4,4,2,1,1,1,2,5,3,3,2,1,1,1,
             5,5,3,1,4,4,2,4,2,1,1,1,5,1,2,4,1,3,4,4,2,1,4,2,1,3,4,3,3,2,3,1,5,
             3,1,1,5,1,2,2,4,4,1,2,3,1,2,1,1,2,1,1,1,2,3,5,5,1,2,3,1,3,5,4,2,1,
             3,3,4)

    fishTable <- c(0, 0, table(dt2), rep(0, 8 - max(dt2)))

    fish <- function(v, d) {
      if (d == 0) {
        return(sum(v, digits = 999))
      } else if (d == 176) {
        s <- sum(v)
      }
      fish(c(v[2:7], v[8] + v[1], v[9:10], v[2]), d - 1)
    }
    s <- fish(fishTable, 256)
  },
  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 With recursion!         1000     0.536    0.004   0.540 0.000536
## 2       Third try         1000     2.537    0.049   2.587 0.002537
## 1       First try         1000    55.870    7.578  63.404 0.055870
```

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) {
```

```r
      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     0.551    0.102   0.653 0.000551
## 1 simple loop         1000     1.795    0.134   1.928 0.001795
## 3 nested loop         1000    35.317   10.041  45.361 0.035317
```