Advent of Code 2021 - Day 3

Matthias Zepper

12/3/2021

Solutions to the Day 3 tasks of the Advent of code

Part 1

You need to use the binary numbers in the diagnostic report to generate two new binary numbers (called the gamma rate and the epsilon rate). The power consumption can then be found by multiplying the gamma rate by the epsilon rate. Each bit in the gamma rate can be determined by finding the most common bit in the corresponding position of all numbers in the diagnostic report. For example, given the following diagnostic report:

The epsilon rate is calculated in a similar way; rather than use the most common bit, the least common bit from each position is used. So, the epsilon rate is 01001, or 9 in decimal. Multiplying the gamma rate (22) by the epsilon rate (9) produces the power consumption, 198.

Reading in the input data

Even though these are binary numbers, we will force them to be read as `character` type values to simplify downstream processing.

```
bincodes <- read.delim("day_3_input.txt",as.is=TRUE,colClasses="character", col.names="errorcode", heade
```

Determine the to rates

`sapply` can apply a function like `strsplit` over each of the status codes and will split every digit into a own string. Afterwards, I am creating a matrix out of the respective single error codes:

```
df_bincodes <- lapply(sapply(bincodes,strsplit,split=""),as.numeric)
df_bincodes <- do.call("rbind",df_bincodes)
```

To figure out either 0 or 1 is more common in each column, I will calculate the mean and test, if it is larger than 0.5:

```
gamma <- colMeans(df_bincodes)>0.5  #Boolean
print(gamma)
```

```
##  [1] FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
```

The Boolean Values can be converted back to numeric representation (FALSE will be interpreted as 0 and TRUE as 1)

```
epsilon <- as.numeric(!gamma)
gamma <- as.numeric(gamma)
print(gamma)
```

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

```
print(epsilon)
```

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

Finally, convert binary to decimal and multiply

```
gamma <- strtoi(paste(gamma,collapse = ""),base=2)
epsilon <- strtoi(paste(epsilon,collapse = ""),base=2)
print(gamma)
```

```
## [1] 1565
```

```
print(epsilon)
```

```
## [1] 2530
```

```
print(epsilon*gamma)
```

```
## [1] 3959450
```

Part 2

Next, you should verify the life support rating, which can be determined by multiplying the oxygen generator rating by the CO2 scrubber rating Both the oxygen generator rating and the CO2 scrubber rating are values that can be found in your diagnostic report - finding them is the tricky part. Both values are located using a similar process that involves filtering out values until only one remains.

Because this is an iterative task, I will need a function that I can apply:

```
elim_bincodes <- function(codearray, bitpos, mode = "oxy") {
  bitmean <- mean(codearray[, bitpos])

  if (bitmean >= 0.5) {
    keepval <- ifelse(mode=="oxy",1,0)
  }
  if (bitmean < 0.5) {
    keepval <- ifelse(mode=="oxy",0,1)
  }

  return(codearray[codearray[, bitpos] == keepval, ])
}
```

Calculate the oxygen generator rating

```
bitpos <- 1
codearray <- df_bincodes

while (dim(codearray)[1] > 1) {
  print(paste("Comparing bit postion", bitpos))
  codearray <- elim_bincodes(codearray, bitpos, mode="oxy")
  print(paste("Currently", dim(codearray)[1], "status codes left"))
  bitpos <- bitpos + 1

  if (all(class(codearray) == "numeric")){
    assign("oxigen_rating", strtoi(paste(codearray, collapse = ""), base = 2))
    print(paste("Found oxygen generator rating:", oxigen_rating))
    break
  }
}
```

```
## [1] "Comparing bit postion 1"
## [1] "Currently 518 status codes left"
## [1] "Comparing bit postion 2"
## [1] "Currently 271 status codes left"
## [1] "Comparing bit postion 3"
## [1] "Currently 150 status codes left"
## [1] "Comparing bit postion 4"
## [1] "Currently 84 status codes left"
## [1] "Comparing bit postion 5"
## [1] "Currently 46 status codes left"
## [1] "Comparing bit postion 6"
## [1] "Currently 23 status codes left"
## [1] "Comparing bit postion 7"
## [1] "Currently 13 status codes left"
## [1] "Comparing bit postion 8"
## [1] "Currently 8 status codes left"
## [1] "Comparing bit postion 9"
## [1] "Currently 5 status codes left"
## [1] "Comparing bit postion 10"
## [1] "Currently 3 status codes left"
## [1] "Comparing bit postion 11"
## [1] "Currently 2 status codes left"
## [1] "Comparing bit postion 12"
## [1] "Currently  status codes left"
## [1] "Found oxygen generator rating: 2039"
```

```
bitpos <- 1
codearray <- df_bincodes

while (dim(codearray)[1] > 1) {
  print(paste("Comparing bit postion", bitpos))
  codearray <- elim_bincodes(codearray, bitpos, mode="CO2")
  print(paste("Currently", dim(codearray)[1], "status codes left"))
  bitpos <- bitpos + 1

  if (all(class(codearray) == "numeric")) {
    assign("carbondiox_scrubber", strtoi(paste(codearray, collapse = ""), base = 2))
    print(paste("Found CO2 scrubber rating:",carbondiox_scrubber))
    break
  }
}
```

```
## [1] "Comparing bit postion 1"
## [1] "Currently 482 status codes left"
## [1] "Comparing bit postion 2"
## [1] "Currently 231 status codes left"
## [1] "Comparing bit postion 3"
## [1] "Currently 108 status codes left"
## [1] "Comparing bit postion 4"
## [1] "Currently 54 status codes left"
## [1] "Comparing bit postion 5"
## [1] "Currently 26 status codes left"
## [1] "Comparing bit postion 6"
## [1] "Currently 12 status codes left"
## [1] "Comparing bit postion 7"
## [1] "Currently 6 status codes left"
## [1] "Comparing bit postion 8"
## [1] "Currently 3 status codes left"
## [1] "Comparing bit postion 9"
## [1] "Currently  status codes left"
## [1] "Found CO2 scrubber rating: 3649"
```

Finally, multiply:

```
oxigen_rating * carbondiox_scrubber
```

```
## [1] 7440311
```