H Math Challenge Problem

Lindsey Alexanian

3/2/22

```r
h_df <- data.frame('one'=c(NA, NA, NA))

vector1 <- c(NA, NA, NA)

h_df <- cbind(h_df, vector1)

h_df <- cbind(h_df, vector1)

colnames(h_df) = c("one", "two", "three")

vector_options <- c(1, 2, 3, 4, 5, 6, 7)

prod_middle <- h_df$one[2]*h_df$two[2]*h_df$three[2]

sum_left_h <- sum(h_df$one[1], h_df$one[2], h_df$one[3])

sum_right_h <- sum(h_df$three[1], h_df$three[2], h_df$three[3])

sum_left_diag <- sum(h_df$one[1], h_df$two[2], h_df$three[3])

sum_right_diag <- sum(h_df$three[1], h_df$two[2], h_df$one[3])

# install package for nPr
install.packages("combinat")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```r
library(combinat)
```

```
##
## Attaching package: 'combinat'
```

```
## The following object is masked from 'package:utils':
##
##     combn
```

```r
# calculate permutations
df_npr <- as.data.frame(permn(vector_options))

# change column names
colnames(df_npr) <- c(1:ncol(df_npr))

# initialize vector of indices in npr vector
```

```r
index_and_product <- data.frame()

# loop through the permutations
for (i in 1:ncol(df_npr)) {

  # assign values to the H data frame
  h_df$one[1] <- df_npr[1, i]
  h_df$one[2] <- df_npr[2, i]
  h_df$one[3] <- df_npr[3, i]
  h_df$two[2] <- df_npr[4, i]
  h_df$three[1] <- df_npr[5, i]
  h_df$three[2] <- df_npr[6, i]
  h_df$three[3] <- df_npr[7, i]

  # calculate product of middle row
  prod_middle <- h_df$one[2]*h_df$two[2]*h_df$three[2]

  # sum of left column
sum_left_h <- sum(h_df$one[1], h_df$one[2], h_df$one[3])

  # sum of right column
sum_right_h <- sum(h_df$three[1], h_df$three[2], h_df$three[3])

  # sum of diagonal from upper left to lower right
sum_left_diag <- sum(h_df$one[1], h_df$two[2], h_df$three[3])

  # sum of diagonal from upper right to lower left
sum_right_diag <- sum(h_df$three[1], h_df$two[2], h_df$one[3])

  # if the sum of the two vertical and diagonals are ALL equal
  if (isTRUE(all(identical(sum_left_h, sum_right_h, num.eq = TRUE), identical(sum_left_diag, sum_right_

    # vector containing index and product
    vector_i_prod <- c(i, prod_middle)

    # add it to the filtered df
    index_and_product <- rbind(index_and_product, vector_i_prod)

  }
}

# rename the columns in the filtered df
colnames(index_and_product) <- c("Index", "Product")

# find minimum product value
minimum_product <- min(index_and_product$Product)

# initialize vector for indices that return the minimum product
indices <- c()

# get indices
for (row in 1:nrow(index_and_product)) {
  if (index_and_product$Product[row] == minimum_product) {
    indices <- append(indices, index_and_product$Index[row])
  }

}

# use indices to return permutation
for (i in indices) {
  print(df_npr[, i])
}
```

```
## [1] 7 1 5 2 6 3 4
## [1] 5 1 7 2 4 3 6
## [1] 4 3 6 2 5 1 7
## [1] 6 3 4 2 7 1 5
```

This is what the possible outcomes look like

```
# use indices to return permutation
for (i in indices) {
  vector_position <- df_npr[, i]
  h_df$one[1] <- vector_position[1]
  h_df$one[2] <- vector_position[2]
  h_df$one[3] <- vector_position[3]
  h_df$two[2] <- vector_position[4]
  h_df$three[1] <- vector_position[5]
  h_df$three[2] <- vector_position[6]
  h_df$three[3] <- vector_position[7]

  print(h_df)

}
```

```
##   one two three
## 1   7  NA     6
## 2   1   2     3
## 3   5  NA     4
##   one two three
## 1   5  NA     4
## 2   1   2     3
## 3   7  NA     6
##   one two three
## 1   4  NA     5
## 2   3   2     1
## 3   6  NA     7
##   one two three
## 1   6  NA     7
## 2   3   2     1
## 3   4  NA     5
```