Functionals(apply, sapply)

Chandana Dasari

22 January 2019

Given a matrix of exam scores for 10 students:

```
set.seed(10)
N=10
cs1 <- rnorm(N, 72, 10)
cs2 <- rnorm(N,65,7)
cs3 <- rnorm(N,80,9)
cs4 <- rnorm(N,55,7)
cs5 <- rnorm(N,61,5)

m <- matrix(c(cs1, cs2, cs3, cs4, cs5), nrow = 10, ncol = 5)
colnames(m) <- c('cs1', 'cs2', 'cs3', 'cs4', 'cs5')
rownames(m) <- paste("Student#", 1:10, sep = '')
m
```

```
##                  cs1      cs2      cs3      cs4      cs5
## Student#1  72.18746 72.71246 74.63320 42.02382 66.43276
## Student#2  70.15747 70.29047 60.33242 54.45438 57.18728
## Student#3  58.28669 63.33237 73.92621 61.77996 56.85669
## Student#4  66.00832 71.91211 60.92845 56.29448 65.17237
## Student#5  74.94545 70.18973 68.61322 45.34039 56.16174
## Student#6  75.89794 65.62543 76.63705 44.95140 60.85592
## Student#7  59.91924 58.31539 73.81200 57.53461 62.16263
## Student#8  68.36324 63.63395 72.15057 42.68639 59.49396
## Student#9  55.73327 71.47865 79.08415 52.72819 57.61193
## Student#10 69.43522 68.38085 77.71598 50.43906 64.27614
```

Generate the following ranked output using apply functions (loops cannot be used).

```
my_rank <- function(x) {
  names(x) <- 1:length(x)
  sorted_x <- sort(x, decreasing = TRUE)
  sort_index <- as.integer(names(sorted_x))
  rank_x <- c()
  rank_x[c(sort_index)] <- 1:length(x)
  rank_x
}

rm <- apply(m, 2, my_rank)
rownames(rm) <- paste("Student#", 1:10, sep = '')
rm
```

```
##             cs1 cs2 cs3 cs4 cs5
## Student#1     3   1   4  10   1
## Student#2     4   4  10   4   8
## Student#3     9   9   5   1   9
## Student#4     7   2   9   3   2
## Student#5     2   5   8   7  10
## Student#6     1   7   3   8   5
## Student#7     8  10   6   2   4
## Student#8     6   8   7   9   6
## Student#9    10   3   1   5   7
## Student#10    5   6   2   6   3
```

Based on this output, calculate the median rank for each student as follows.

```
rm <- cbind(rm, median=apply(rm, 1, median))
rm
```

```
##             cs1 cs2 cs3 cs4 cs5 median
## Student#1     3   1   4  10   1      3
## Student#2     4   4  10   4   8      4
## Student#3     9   9   5   1   9      9
## Student#4     7   2   9   3   2      3
## Student#5     2   5   8   7  10      7
## Student#6     1   7   3   8   5      5
## Student#7     8  10   6   2   4      6
## Student#8     6   8   7   9   6      7
## Student#9    10   3   1   5   7      5
## Student#10    5   6   2   6   3      5
```