Lab 1

Osman Khan

11:59PM February 1

You should have RStudio installed to edit this file. You will write code in places marked "TO-DO" to complete the problems. Most of this will be a pure programming assignment but there are some questions that instead ask you to "write a few sentences". This is a W class! The tools for the solutions to these problems can be found in the class practice lectures. I prefer you to use the methods I taught you. If you google and find esoteric code you don't understand, this doesn't do you too much good.

To "hand in" the homework, you should first download this file. The best way to do this is by cloning the class repository then copying this file from the folder of that clone into the folder that is your personal class repository. Then do the assignment by filling in the TO-DO's. After you're done, compile this file into a PDF (use the "knit to PDF" button on the submenu above). This PDF will include output of your code. Then push the PDF and this Rmd file by the deadline to your github repository in a directory called "labs".

Basic R Skills

- Print out the numerical constant pi with ten digits after the decimal point using the internal constant `pi`.

```
#the default in R is 7 decimal places, so we increase significant digits to 11
options(digits = 11)
pi
```

```
## [1] 3.1415926536
```

- Sum up the first 103 terms of the series $1 + 1/2 + 1/4 + 1/8 + \ldots$

```
#we create a vector with dim[103], and then sum all terms:
sum(2^(-(0:102)))
```

```
## [1] 2
```

- Find the product of the first 37 terms in the sequence $1/3, 1/6, 1/9 \ldots$

```
#we create a vector with dim[37], and then multiply all terms
prod(1/(3*(1:37)))
```

```
## [1] 1.613528728e-61
```

- Find the product of the first 387 terms of `1 * 1/2 * 1/4 * 1/8 * ...`

```
#we create a vector with dim[387], and then multiply all terms
prod(2^(-(0:386)))
```

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

Is this answer *exactly* correct?

No

- Figure out a means to express the answer more exactly. Not compute exactly, but express more exactly.

```
#we can express each term of the vector as log base ten, then, we use properties
#of log to sum all log'ed terms
sum(log10(2^(-(0:386))))
```

```
## [1] -22484.231406
```

- Create the sequence `x = [Inf, 20, 18, ..., -20]`.

```
#we concatenate Inf with the sequence from 20 to -20 by -2
x=c(1/0,seq(20,-20,by=-2))
```

Create the sequence `x = [log_3(Inf), log_3(100), log_3(98), ... log_3(-20)]`.

```
#we concatenate Inf with the sequence from 100 to -20 by -2, then we take log
#base 3 of all terms
x=log(c(Inf,seq(100,-20,by=-2)),base=3)
```

```
## Warning: NaNs produced
```

Comment on the appropriateness of the non-numeric values NAN and -Inf.

We are getting NaN for 0 and -Inf for negative inputs for log_3. This is reasonable.

- Create a vector of booleans where the entry is true if `x[i]` is positive and finite.

```
x>0&is.finite(x)
```

```
##  [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [25]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [37]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [49]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [61] FALSE FALSE
```

- Locate the indices of the non-real numbers in this vector. Hint: use the `which` function. Don't hesitate to use the documentation via `?which`.

```
#?which
which(is.infinite(x)|is.nan(x))
```

```
##  [1]  1 52 53 54 55 56 57 58 59 60 61 62
```

- Locate the indices of the infinite quantities in this vector.

```
which(is.infinite(x))
```

```
## [1]  1 52
```

- Locate the indices of the min and max in this vector. Hint: use the `which.min` and `which.max` functions.

```
#?which.max
which.min(ifelse(is.infinite(x), NA, x))
```

```
## [1] 51
```

```
which.max(ifelse(is.infinite(x), NA, x))
```

```
## [1] 2
```

```
#second option
x_without_infs = ifelse(is.infinite(x), NA, x)
which.min(x_without_infs)
```

```
## [1] 51
```

```
which.max(x_without_infs)
```

```
## [1] 2
```

```
rm(x_without_infs)
#?Find
```

- Count the number of unique values in `x`.

```
length(unique(x))
```

```
## [1] 53
```

- Cast `x` to a factor. Do the number of levels make sense?

```
as.factor(x)
```

```
##  [1] Inf               4.19180654857877  4.1734172518943   4.15464876785729
##  [5] 4.13548512895119  4.11590933734319  4.09590327428938  4.07544759935851
##  [9] 4.05452163806914  4.03310325630434  4.01116871959141  3.98869253500376
## [13] 3.96564727304425  3.94200336638929  3.91772888178973  3.89278926071437
## [17] 3.86714702345081  3.84076143030548  3.81358809221559  3.78557852142874
## [21] 3.75667961082847  3.72683302786084  3.69597450568212  3.66403300987579
## [25] 3.63092975357146  3.59657702661571  3.56087679500731  3.52371901428583
## [29] 3.48497958377173  3.44451784578705  3.40217350273288  3.3577627814323
## [33] 3.31107361281783  3.26185950714291  3.20983167673402  3.15464876785729
## [37] 3.09590327428938  3.03310325630434  2.96564727304425  2.89278926071437
## [41] 2.8135880922156   2.72683302786084  2.63092975357146  2.52371901428583
## [45] 2.40217350273288  2.26185950714291  2.09590327428938  1.89278926071437
## [49] 1.63092975357146  1.26185950714291  0.630929753571457 -Inf
## [53] NaN               NaN               NaN               NaN
## [57] NaN               NaN               NaN               NaN
## [61] NaN               NaN
## 53 Levels: -Inf 0.630929753571457 1.26185950714291 ... NaN
```

- Cast `x` to integers. What do we learn about R's infinity representation in the integer data type?

```
as.integer(x)
```

```
## Warning: NAs introduced by coercion to integer range
```

```
##  [1] NA  4  4  4  4  4  4  4  4  4  4  3  3  3  3  3  3  3  3  3  3  3  3  3
## [26]  3  3  3  3  3  3  3  3  3  3  3  3  2  2  2  2  2  2  2  2  2  1  1  1
## [51]  0 NA NA NA NA NA NA NA NA NA NA NA
```

```
#positive and negative infinity are represented by NA in the integer data type.
```

- Use `x` to create a new vector `y` containing only the real numbers in x.

```
y=x[is.finite(x)]
```

- Use the left rectangle method to numerically integrate x^2 from 0 to 1 with rectangle width size 1e-6.

```
delta_x=1e-6
sum((seq(0,1-delta_x,by=delta_x))^2)*delta_x
```

```
## [1] 0.33333283333
```

- Calculate the average of 100 realizations of standard Bernoullis in one line using the `sample` function.

```
#?sample
x_bern=c(0,1)
mean(sample(x_bern,size=100, replace=TRUE))
```

```
## [1] 0.59
```

- Calculate the average of 500 realizations of Bernoullis with p = 0.9 in one line using the `sample` and `mean` functions.

```
mean(sample(c(0,1),size=500, replace=TRUE, prob = c(0.1, 0.9)))
```

```
## [1] 0.914
```

- Calculate the average of 1000 realizations of Bernoullis with p = 0.9 in one line using `rbinom`.

```
mean(rbinom(n=1000,size=1,prob=0.9))
```

```
## [1] 0.91
```

- Let `n = 50`. Create a n x n matrix `R` of exactly 50% entries 0's, 25% 1's 25% 2's. These values should be in random locations.

```
vec = sample(c(rep(0,1250),rep(1,625),rep(2,625)))
R=matrix(vec,nrow=50,ncol=50)

table(c(R))
```

```
##
##    0    1    2
## 1250  625  625
```

```
#?matrix
```

- Randomly punch holes (i.e. `NA`) values in this matrix so that an each entry is missing with probability 30%.

```
R[runif(2500) <.3] = NA
R
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
##  [1,]    0   NA    2    0   NA   NA    2    0    1     0     1     2     2
##  [2,]    2    0    2   NA    0   NA    1    1    0    NA     2    NA     2
##  [3,]    1   NA   NA    0   NA   NA   NA    2   NA    NA    NA     0     0
##  [4,]    0    2    0    2    2    0    1    0    2     1    NA     1    NA
##  [5,]    1    0    2   NA    0    1    2   NA    1     0     2     0     0
##  [6,]    0    0    0    2    0    0   NA   NA    0     2     1     0    NA
##  [7,]   NA    0    1    2    0   NA   NA    0    1    NA    NA     0    NA
##  [8,]    1   NA    2    0    0    2    2    1    0     0     0    NA     2
##  [9,]    0    2    0    2    0   NA    0    2    0    NA     2     0     2
## [10,]    0    0    0    0    0    1   NA    0    2     0     1    NA    NA
## [11,]   NA    2    2    1    0    2   NA    0   NA     1     1    NA     2
## [12,]    0   NA    2    0   NA    0    2    1    0     2     2     1     0
## [13,]    0    0    1   NA    0    0    2   NA   NA     0     1    NA     0
## [14,]    2   NA    1   NA   NA    1   NA    0    2     2    NA     2     2
## [15,]    2   NA   NA   NA   NA    0    2   NA    1     1     0    NA    NA
## [16,]    2    0   NA    1    1    0    0   NA    0     0     0    NA     1
## [17,]    1    1    2    1   NA   NA    0    2   NA    NA     0     0     1
## [18,]    0    0    0   NA    0    0    1    0    0     0    NA    NA     2
## [19,]   NA    0   NA    1   NA   NA    2    0    1     0     0    NA     2
```

```
## [20,]   NA   NA   NA    1   NA   NA    1    0    0     0     2    NA    NA
## [21,]    0    1   NA    0    0    0   NA    1   NA     0    NA    NA    NA
## [22,]   NA    2   NA    0    0    1    1    0    0     2     2     1     1
## [23,]    0    0   NA    0    2    1   NA    0   NA     0     0     0     1
## [24,]    2    0    2    0    1    2    2    0    1     1    NA    NA     1
## [25,]   NA    1    0   NA    1    0    1    0    2     1     0     2    NA
## [26,]    0    2    2    2    0    0   NA    0    0    NA     0    NA     0
## [27,]    0    0   NA    0    0    1   NA   NA   NA     2    NA     2     0
## [28,]    1    1   NA   NA    1    2   NA    0    2    NA    NA     2     0
## [29,]    2    2    0    0    0   NA   NA    1   NA     0    NA     2    NA
## [30,]    0   NA    0    1    0    2   NA    0    1     0     2    NA     0
## [31,]   NA    0    0    0   NA   NA    0   NA   NA     2     2     1     0
## [32,]    0   NA    0    2    2    1    1    0    2     0     2    NA     1
## [33,]   NA   NA    2   NA    1    0    0    0   NA     0    NA     1     0
## [34,]    1    1   NA    1   NA    1    0   NA   NA     1     0     0     0
## [35,]    2    0    1   NA    2    2    0    2    0     2     0     1     0
## [36,]   NA    1    2    0    0    0   NA    0    2     0    NA     0     0
## [37,]    1    0    2   NA   NA    2   NA    0   NA    NA    NA     0     2
## [38,]    1    0    2    2   NA    0   NA    0    1    NA    NA    NA    NA
## [39,]    2   NA    2   NA   NA    0   NA    2   NA     0     0     0     0
## [40,]   NA   NA   NA    0   NA   NA    0    0    1    NA     0    NA    NA
## [41,]    0    1   NA    0   NA    0   NA   NA    0    NA     0     0     2
## [42,]    2    0    0   NA    0    0    0    2   NA     0     0     0     0
## [43,]   NA    1    1    0    1    2    0    1   NA     0     1     1     2
## [44,]   NA    2   NA    2    0    0    1   NA    0     1    NA     0     0
## [45,]    0    0   NA   NA   NA   NA   NA   NA   NA     0     2     0     0
## [46,]    0    0    2    0   NA    1    0    0    0     2     1     2     1
## [47,]    0    2    2   NA    2    0    0    2    0     1     0     1    NA
## [48,]    1    1   NA    0    0    2    0    1   NA    NA     0     1     1
## [49,]    0   NA    0   NA    2    2    1    0    0    NA     0     2     0
## [50,]    2   NA   NA    1   NA    0    1   NA    1     2     0    NA     0
##       [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25]
##  [1,]     0     1    NA     1    NA    NA    NA    NA    NA     0     0    NA
##  [2,]     0     1     0     2     0    NA    NA     2     0     2    NA     1
##  [3,]     2     0     0     1     0     0    NA    NA    NA     2    NA     0
##  [4,]     1     1     1    NA    NA     1     0    NA     1     0     1     2
##  [5,]     1     2    NA     0    NA     0    NA     0     0    NA     2     1
##  [6,]    NA    NA    NA    NA    NA    NA     0    NA     1     0    NA     0
##  [7,]     2    NA    NA    NA     1    NA     0    NA     2    NA     1     2
##  [8,]     2     0    NA     0    NA     0     0    NA    NA     0     0     1
##  [9,]     2    NA     2    NA     1    NA     0     0     1    NA     0     0
## [10,]     1     2     0     2     1     1     1     0     0     0    NA     0
## [11,]     0     1     2    NA    NA    NA     0     0     2    NA    NA     0
## [12,]     1    NA    NA     0     1    NA     0    NA     0     0     0    NA
## [13,]     2     0     1    NA     2     1     0     0    NA     1     0    NA
## [14,]     1     0     0     0     0     2     0     1     1     0     0    NA
## [15,]    NA    NA     2    NA     2     1    NA    NA    NA     1    NA     1
## [16,]    NA     2    NA     0     0     1     0     0     0    NA     0     2
## [17,]     2     1    NA     1     2    NA     0     0     0     1     2    NA
## [18,]     1     2     2     2    NA    NA     0    NA    NA    NA     1     0
## [19,]     0     0     2    NA     2     1     0     1     1     1    NA     0
## [20,]     0     0    NA     1     0    NA     1     0    NA     1     0     0
## [21,]     0    NA     2    NA     0     0    NA     0     0     1     0     0
## [22,]    NA    NA     1     0    NA    NA     1    NA    NA     0    NA     0
```

```
## [23,]    NA     1     2    NA    NA     0     0     0    NA     1     0     0
## [24,]     0    NA     2     2     0     1     1    NA     0     1     2     2
## [25,]     0    NA     2    NA     0     2     1    NA     0    NA     0    NA
## [26,]    NA     1     0     0     0    NA     0     0     0     0     0    NA
## [27,]    NA     0    NA     1     1    NA     1     0     1     0     2     1
## [28,]     1     2     0     1     0     1     2     1     0    NA    NA    NA
## [29,]    NA     0    NA     0    NA     2     0    NA     0     0    NA     2
## [30,]     1    NA     2     0    NA    NA     0     0    NA     2     0    NA
## [31,]     0     2     1    NA     2     0     0     0     0     2     1     0
## [32,]    NA     1    NA     0     0    NA    NA     1    NA    NA    NA     0
## [33,]     0    NA     2    NA     1     2     0     0     1     0    NA     2
## [34,]     2     0     0     0     0     0     1    NA     1    NA    NA     0
## [35,]     1     1     2     0    NA     2    NA     2     2     0    NA    NA
## [36,]     1    NA     0     0     1     1     2     0    NA    NA     1    NA
## [37,]     2     0     0     0     0     0    NA     0    NA     1     1     0
## [38,]     0     1     0     0     0     1    NA     0     0     1     0     0
## [39,]     1    NA     0     2     0     2    NA    NA     1     2     1     2
## [40,]    NA     2    NA     2     1     1    NA     0     0     1    NA    NA
## [41,]     2     1    NA     0     1     2    NA    NA     0     2    NA     2
## [42,]     2    NA     2    NA     0     0    NA     1    NA     2     1     0
## [43,]     2    NA     0    NA    NA     2    NA     0    NA     2     2     1
## [44,]    NA     1     0     0    NA     0    NA    NA     0     0     0     1
## [45,]     0     1     1     0     1     1    NA     1     0     0    NA    NA
## [46,]     2    NA     0     1    NA     2     0     2     0    NA     1     1
## [47,]     2    NA    NA     0    NA    NA    NA    NA    NA     0     0    NA
## [48,]     0     1    NA     0     0     0     0     0     0     0     1     2
## [49,]    NA     2     0     0     0     0    NA     1     1     0    NA     0
## [50,]    NA     1    NA     2     2     0     2     1     2     0     0    NA
##       [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35] [,36] [,37]
##  [1,]    NA     1    NA     1     0    NA    NA     0     1    NA    NA     1
##  [2,]     2     1     1    NA     0     1    NA     0     1     0     0     2
##  [3,]     0     0     2     1     1    NA     2    NA     1     0     2     2
##  [4,]    NA    NA    NA     0     0     0     2     0     0    NA     2     0
##  [5,]     0     0    NA    NA     2    NA     0     1    NA    NA     0     2
##  [6,]     1     0    NA     1    NA    NA     0     0     0    NA     0    NA
##  [7,]    NA     2     0    NA     0    NA     0     2     2     0     2     2
##  [8,]     0     0    NA     1    NA     1     0     2     1     0     0    NA
##  [9,]    NA     0     0     0     0    NA     0     2     2    NA    NA    NA
## [10,]    NA     0     2    NA     0     1     2     0     0    NA     0     0
## [11,]    NA     0     0    NA     0     0     0     1    NA     1     0     0
## [12,]     2     2     1    NA     2     1     0    NA     2     0     2     1
## [13,]     1     2     0     2    NA    NA    NA     1     2     2    NA     0
## [14,]     2     2     2     1     0     2    NA    NA     0     0    NA    NA
## [15,]     0     1    NA     2     1     2     2     1     2     1     2     0
## [16,]     0     0     1    NA     0    NA     2     0     1     0    NA     1
## [17,]    NA     1    NA     0     0    NA     2     2    NA     0     1     1
## [18,]    NA     2    NA     0     1    NA     0    NA     2    NA    NA     0
## [19,]    NA     0     0     0     2     0    NA     0     0     0     0     2
## [20,]     0    NA     0     0     1    NA    NA     0    NA    NA     1    NA
## [21,]    NA     2     2     0     0    NA     2     2     0     0    NA     2
## [22,]     1     1     2    NA     1     1     0     0     0     0     1     1
## [23,]     2    NA     1     0     1    NA    NA    NA     0     0     1     0
## [24,]    NA     0     2     2    NA    NA     2     0    NA     0    NA     2
## [25,]     2    NA     0     2     1     0     1    NA     0    NA     0     1
```

```
## [26,]     0     2     0     0     1    NA    NA     2     2     1    NA     0
## [27,]     1     0     2    NA     0     2     2     2     2    NA     0    NA
## [28,]    NA     0    NA     0     2     1     1     0     1     2     1    NA
## [29,]     0     0    NA     1    NA    NA     2     0     0     0     1     1
## [30,]     2     2     1     0    NA     2    NA    NA     0     2     1     1
## [31,]    NA     0     1    NA     0    NA     0     1     1     1     1     1
## [32,]    NA     0     0     0    NA     0    NA     0     2     2     1     2
## [33,]     2     0     1     0    NA     0    NA     2    NA    NA     1    NA
## [34,]     0     0    NA    NA     1    NA     1     0     1    NA     2     2
## [35,]     0     2     0    NA     2     0     0    NA     1     0     0     2
## [36,]    NA    NA     0     2     0    NA    NA     0    NA     1     2     0
## [37,]     2     0     0    NA    NA    NA     0     1     0     2     0     0
## [38,]    NA     0     0    NA     1     0     0     0     0     0    NA     0
## [39,]     1    NA    NA     0     1    NA    NA     2     0    NA     1     2
## [40,]     0     2     0     0     0     1    NA    NA     0     1    NA    NA
## [41,]     1    NA    NA     2     0    NA    NA     1    NA    NA     0     1
## [42,]     1     0    NA    NA     1     1    NA     1     0     0     1    NA
## [43,]     0     0     2     1     0     2     2    NA    NA    NA    NA     0
## [44,]     1     0     0     0    NA     0     1     1     0     0     0     0
## [45,]    NA     2     1     0     0     0     0     0    NA     0     2     0
## [46,]     2     0    NA     0     0     0     0     1     0     2     2     0
## [47,]     0     0     2     0     0     1    NA    NA     1    NA     0     0
## [48,]     0    NA     2     2     1    NA     0     1    NA     0    NA     2
## [49,]     0     2     1     1    NA    NA     2     1     2     0     1    NA
## [50,]     2     1     0    NA     2     0     0     1     1    NA     1     2
##       [,38] [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46] [,47] [,48] [,49]
##  [1,]     0     1    NA     1     0     2     1     2     0     1     0     2
##  [2,]     0     0     1     2    NA    NA    NA     0    NA     0     2     1
##  [3,]     1     0     0     0     1     0    NA     0     0     2     1     0
##  [4,]     2     2    NA    NA     0     1    NA     2     0    NA     2     1
##  [5,]    NA     1    NA     0     0     2    NA     1    NA     1    NA     0
##  [6,]    NA     0    NA     0     1     1    NA    NA    NA     2     1     1
##  [7,]     1     1     2     2     1     1     2     0    NA    NA     2     0
##  [8,]     0    NA    NA     2     1    NA     0    NA    NA     0     1    NA
##  [9,]    NA    NA     1     0     0     0     1    NA    NA    NA     1     0
## [10,]    NA     2     0     2     0     0    NA     0    NA     2    NA    NA
## [11,]     0    NA     1     0     1     2     0     1     0     0    NA    NA
## [12,]    NA    NA     2     1    NA    NA     0     0     0     1     0     0
## [13,]     0     0     1     1     1     0     1     2     1     0    NA    NA
## [14,]     2    NA     0     1     0     0    NA     1     2     0     0    NA
## [15,]     2     2     2    NA     0     0     1     2     0    NA     2     0
## [16,]    NA     0     2     0     0     2    NA    NA     0     2     2     0
## [17,]     0     1     0     2     0     0     2     0     0     0    NA     2
## [18,]    NA     2     0    NA     0     0     0     0     2    NA     0     2
## [19,]    NA    NA    NA     0     0     1    NA     0     0    NA     0    NA
## [20,]     0    NA    NA    NA     0     0     0     1     0     1     0     2
## [21,]     0     2     1    NA    NA    NA     2     0     0     2     1     0
## [22,]     0     0    NA     0     1     1     0    NA     2     0     1     2
## [23,]     0    NA    NA    NA     2     2     2     1    NA    NA     0     1
## [24,]    NA     0    NA     0    NA     0     2     2     0    NA     2    NA
## [25,]     0     2    NA     2    NA    NA     1     0    NA     0     0     1
## [26,]     1     0    NA    NA     0     0     0    NA     0    NA    NA     0
## [27,]    NA    NA    NA    NA     1    NA     2     1     2     0    NA     2
## [28,]     2    NA    NA     1     0    NA     0     1     0    NA     0     0
```

```
## [29,]     0     2     0    NA     0     1    NA     2     0    NA    NA     1
## [30,]    NA    NA     2     0    NA    NA    NA    NA    NA    NA    NA    NA
## [31,]     1     2     2     1     0    NA    NA     0     0     2     0     1
## [32,]    NA     1    NA    NA    NA     1     1     2    NA     2     2     2
## [33,]     1     1    NA     1     2     1     0    NA     0     0     1     0
## [34,]     0     0    NA    NA    NA     1     0     0     2     0     2    NA
## [35,]    NA    NA     0     0    NA     1     1     2     0    NA     0     0
## [36,]    NA     0    NA     0    NA     2    NA     0     0     0     0    NA
## [37,]    NA     0     0     0     0    NA    NA     2    NA     0    NA     2
## [38,]     2    NA    NA     0    NA     0     2     0     0     1     0     1
## [39,]     0     0     2     2     0    NA    NA    NA    NA     1     0     2
## [40,]    NA    NA     0     1    NA     0     1     0    NA    NA     2     1
## [41,]     1     0    NA    NA     0    NA     1     0     2     0    NA     2
## [42,]     2    NA     1     2    NA    NA     0     0    NA     0    NA    NA
## [43,]    NA     0     2     0    NA     2     2     0     2     0    NA    NA
## [44,]    NA     1    NA     1     1     0     1     0    NA     2     0     0
## [45,]     0    NA    NA     0     0    NA     2    NA     0    NA     0     0
## [46,]     2     2     0    NA     2    NA     0     2     0     1     0     0
## [47,]     0     2    NA     2     1    NA    NA     0    NA    NA     0    NA
## [48,]     2     2     0     2     0     2    NA     2     2     1     2     1
## [49,]    NA     0     1     0    NA     1     1     1     1     0     0    NA
## [50,]    NA    NA    NA    NA     2     2     0     2     2    NA     2    NA
##       [,50]
##  [1,]    NA
##  [2,]     2
##  [3,]     0
##  [4,]     2
##  [5,]     2
##  [6,]     2
##  [7,]     0
##  [8,]     0
##  [9,]     1
## [10,]    NA
## [11,]     1
## [12,]     2
## [13,]    NA
## [14,]    NA
## [15,]    NA
## [16,]     0
## [17,]     0
## [18,]     0
## [19,]     0
## [20,]     1
## [21,]     2
## [22,]     1
## [23,]     1
## [24,]    NA
## [25,]     1
## [26,]     1
## [27,]     1
## [28,]     0
## [29,]     0
## [30,]    NA
## [31,]     2
```

```
## [32,]     2
## [33,]     0
## [34,]     1
## [35,]     0
## [36,]     1
## [37,]     0
## [38,]     2
## [39,]    NA
## [40,]     2
## [41,]     2
## [42,]     0
## [43,]     0
## [44,]     1
## [45,]     2
## [46,]     2
## [47,]     2
## [48,]    NA
## [49,]     2
## [50,]    NA
```

```
table(c(R))
```

```
##
##   0   1   2
## 861 437 439
```

- Sort the rows in matrix `R` by the largest row sum to lowest. Be careful about the NA's!

```
R[order(rowSums(R,na.rm=TRUE),decreasing=TRUE),]
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
##  [1,]    2   NA   NA   NA   NA    0    2   NA    1     1     0    NA    NA
##  [2,]    2    0    2    0    1    2    2    0    1     1    NA    NA     1
##  [3,]    2   NA   NA    1   NA    0    1   NA    1     2     0    NA     0
##  [4,]    0    0    2    0   NA    1    0    0    0     2     1     2     1
##  [5,]    0    2    0    2    2    0    1    0    2     1    NA     1    NA
##  [6,]    1    1   NA    0    0    2    0    1   NA    NA     0     1     1
##  [7,]    2    0    2   NA    0   NA    1    1    0    NA     2    NA     2
##  [8,]   NA    0    1    2    0   NA   NA    0    1    NA    NA     0    NA
##  [9,]   NA    1    1    0    1    2    0    1   NA     0     1     1     2
## [10,]    0   NA    0    2    2    1    1    0    2     0     2    NA     1
## [11,]    2    0    1   NA    2    2    0    2    0     2     0     1     0
## [12,]    2   NA    1   NA   NA    1   NA    0    2     2    NA     2     2
## [13,]    0    0   NA    0    0    1   NA   NA   NA     2    NA     2     0
## [14,]    0   NA    2    0   NA    0    2    1    0     2     2     1     0
## [15,]    1    1    2    1   NA   NA    0    2   NA    NA     0     0     1
## [16,]    2   NA    2   NA   NA    0   NA    2   NA     0     0     0     0
## [17,]   NA    0    0    0   NA   NA    0   NA   NA     2     2     1     0
## [18,]    1    1   NA   NA    1    2   NA    0    2    NA    NA     2     0
## [19,]    0    0    1   NA    0    0    2   NA   NA     0     1    NA     0
## [20,]   NA    2   NA    0    0    1    1    0    0     2     2     1     1
## [21,]    0   NA    0   NA    2    2    1    0    0    NA     0     2     0
## [22,]    1    0    2   NA    0    1    2   NA    1     0     2     0     0
## [23,]   NA    1    0   NA    1    0    1    0    2     1     0     2    NA
## [24,]    0   NA    2    0   NA   NA    2    0    1     0     1     2     2
## [25,]    0    1   NA    0   NA    0   NA   NA    0    NA     0     0     2
```

```
## [26,]    0    1   NA    0    0    0   NA    1   NA     0    NA    NA    NA
## [27,]   NA   NA    2   NA    1    0    0    0   NA     0    NA     1     0
## [28,]    1   NA   NA    0   NA   NA   NA    2   NA    NA    NA     0     0
## [29,]    0    2    0    2    0   NA    0    2    0    NA     2     0     2
## [30,]   NA    2    2    1    0    2   NA    0   NA     1     1    NA     2
## [31,]    0   NA    0    1    0    2   NA    0    1     0     2    NA     0
## [32,]    0    0    0    0    0    1   NA    0    2     0     1    NA    NA
## [33,]    2    0   NA    1    1    0    0   NA    0     0     0    NA     1
## [34,]    0    2    2   NA    2    0    0    2    0     1     0     1    NA
## [35,]    1   NA    2    0    0    2    2    1    0     0     0    NA     2
## [36,]    0    0    0   NA    0    0    1    0    0     0    NA    NA     2
## [37,]    0    0   NA    0    2    1   NA    0   NA     0     0     0     1
## [38,]    2    2    0    0    0   NA   NA    1   NA     0    NA     2    NA
## [39,]    1    1   NA    1   NA    1    0   NA   NA     1     0     0     0
## [40,]    2    0    0   NA    0    0    0    2   NA     0     0     0     0
## [41,]    1    0    2   NA   NA    2   NA    0   NA    NA    NA     0     2
## [42,]   NA    0   NA    1   NA   NA    2    0    1     0     0    NA     2
## [43,]   NA    1    2    0    0    0   NA    0    2     0    NA     0     0
## [44,]   NA   NA   NA    0   NA   NA    0    0    1    NA     0    NA    NA
## [45,]    1    0    2    2   NA    0   NA    0    1    NA    NA    NA    NA
## [46,]   NA    2   NA    2    0    0    1   NA    0     1    NA     0     0
## [47,]    0    2    2    2    0    0   NA    0    0    NA     0    NA     0
## [48,]    0    0    0    2    0    0   NA   NA    0     2     1     0    NA
## [49,]    0    0   NA   NA   NA   NA   NA   NA   NA     0     2     0     0
## [50,]   NA   NA   NA    1   NA   NA    1    0    0     0     2    NA    NA
##       [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25]
##  [1,]    NA    NA     2    NA     2     1    NA    NA    NA     1    NA     1
##  [2,]     0    NA     2     2     0     1     1    NA     0     1     2     2
##  [3,]    NA     1    NA     2     2     0     2     1     2     0     0    NA
##  [4,]     2    NA     0     1    NA     2     0     2     0    NA     1     1
##  [5,]     1     1     1    NA    NA     1     0    NA     1     0     1     2
##  [6,]     0     1    NA     0     0     0     0     0     0     0     1     2
##  [7,]     0     1     0     2     0    NA    NA     2     0     2    NA     1
##  [8,]     2    NA    NA    NA     1    NA     0    NA     2    NA     1     2
##  [9,]     2    NA     0    NA    NA     2    NA     0    NA     2     2     1
## [10,]    NA     1    NA     0     0    NA    NA     1    NA    NA    NA     0
## [11,]     1     1     2     0    NA     2    NA     2     2     0    NA    NA
## [12,]     1     0     0     0     0     2     0     1     1     0     0    NA
## [13,]    NA     0    NA     1     1    NA     1     0     1     0     2     1
## [14,]     1    NA    NA     0     1    NA     0    NA     0     0     0    NA
## [15,]     2     1    NA     1     2    NA     0     0     0     1     2    NA
## [16,]     1    NA     0     2     0     2    NA    NA     1     2     1     2
## [17,]     0     2     1    NA     2     0     0     0     0     2     1     0
## [18,]     1     2     0     1     0     1     2     1     0    NA    NA    NA
## [19,]     2     0     1    NA     2     1     0     0    NA     1     0    NA
## [20,]    NA    NA     1     0    NA    NA     1    NA    NA     0    NA     0
## [21,]    NA     2     0     0     0     0    NA     1     1     0    NA     0
## [22,]     1     2    NA     0    NA     0    NA     0     0    NA     2     1
## [23,]     0    NA     2    NA     0     2     1    NA     0    NA     0    NA
## [24,]     0     1    NA     1    NA    NA    NA    NA    NA     0     0    NA
## [25,]     2     1    NA     0     1     2    NA    NA     0     2    NA     2
## [26,]     0    NA     2    NA     0     0    NA     0     0     1     0     0
## [27,]     0    NA     2    NA     1     2     0     0     1     0    NA     2
## [28,]     2     0     0     1     0     0    NA    NA    NA     2    NA     0
```

```
## [29,]     2    NA     2    NA     1    NA     0     0     1    NA     0     0
## [30,]     0     1     2    NA    NA    NA     0     0     2    NA    NA     0
## [31,]     1    NA     2     0    NA    NA     0     0    NA     2     0    NA
## [32,]     1     2     0     2     1     1     1     0     0     0    NA     0
## [33,]    NA     2    NA     0     0     1     0     0     0    NA     0     2
## [34,]     2    NA    NA     0    NA    NA    NA    NA    NA     0     0    NA
## [35,]     2     0    NA     0    NA     0     0    NA    NA     0     0     1
## [36,]     1     2     2     2    NA    NA     0    NA    NA    NA     1     0
## [37,]    NA     1     2    NA    NA     0     0     0    NA     1     0     0
## [38,]    NA     0    NA     0    NA     2     0    NA     0     0    NA     2
## [39,]     2     0     0     0     0     0     1    NA     1    NA    NA     0
## [40,]     2    NA     2    NA     0     0    NA     1    NA     2     1     0
## [41,]     2     0     0     0     0     0    NA     0    NA     1     1     0
## [42,]     0     0     2    NA     2     1     0     1     1     1    NA     0
## [43,]     1    NA     0     0     1     1     2     0    NA    NA     1    NA
## [44,]    NA     2    NA     2     1     1    NA     0     0     1    NA    NA
## [45,]     0     1     0     0     0     1    NA     0     0     1     0     0
## [46,]    NA     1     0     0    NA     0    NA    NA     0     0     0     1
## [47,]    NA     1     0     0     0    NA     0     0     0     0     0    NA
## [48,]    NA    NA    NA    NA    NA    NA     0    NA     1     0    NA     0
## [49,]     0     1     1     0     1     1    NA     1     0     0    NA    NA
## [50,]     0     0    NA     1     0    NA     1     0    NA     1     0     0
##       [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35] [,36] [,37]
##  [1,]     0     1    NA     2     1     2     2     1     2     1     2     0
##  [2,]    NA     0     2     2    NA    NA     2     0    NA     0    NA     2
##  [3,]     2     1     0    NA     2     0     0     1     1    NA     1     2
##  [4,]     2     0    NA     0     0     0     0     1     0     2     2     0
##  [5,]    NA    NA    NA     0     0     0     2     0     0    NA     2     0
##  [6,]     0    NA     2     2     1    NA     0     1    NA     0    NA     2
##  [7,]     2     1     1    NA     0     1    NA     0     1     0     0     2
##  [8,]    NA     2     0    NA     0    NA     0     2     2     0     2     2
##  [9,]     0     0     2     1     0     2     2    NA    NA    NA    NA     0
## [10,]    NA     0     0     0    NA     0    NA     0     2     2     1     2
## [11,]     0     2     0    NA     2     0     0    NA     1     0     0     2
## [12,]     2     2     2     1     0     2    NA    NA     0     0    NA    NA
## [13,]     1     0     2    NA     0     2     2     2     2    NA     0    NA
## [14,]     2     2     1    NA     2     1     0    NA     2     0     2     1
## [15,]    NA     1    NA     0     0    NA     2     2    NA     0     1     1
## [16,]     1    NA    NA     0     1    NA    NA     2     0    NA     1     2
## [17,]    NA     0     1    NA     0    NA     0     1     1     1     1     1
## [18,]    NA     0    NA     0     2     1     1     0     1     2     1    NA
## [19,]     1     2     0     2    NA    NA    NA     1     2     2    NA     0
## [20,]     1     1     2    NA     1     1     0     0     0     0     1     1
## [21,]     0     2     1     1    NA    NA     2     1     2     0     1    NA
## [22,]     0     0    NA    NA     2    NA     0     1    NA    NA     0     2
## [23,]     2    NA     0     2     1     0     1    NA     0    NA     0     1
## [24,]    NA     1    NA     1     0    NA    NA     0     1    NA    NA     1
## [25,]     1    NA    NA     2     0    NA    NA     1    NA    NA     0     1
## [26,]    NA     2     2     0     0    NA     2     2     0     0    NA     2
## [27,]     2     0     1     0    NA     0    NA     2    NA    NA     1    NA
## [28,]     0     0     2     1     1    NA     2    NA     1     0     2     2
## [29,]    NA     0     0     0     0    NA     0     2     2    NA    NA    NA
## [30,]    NA     0     0    NA     0     0     0     1    NA     1     0     0
## [31,]     2     2     1     0    NA     2    NA    NA     0     2     1     1
```

```
## [32,]    NA     0     2    NA     0     1     2     0     0    NA     0     0
## [33,]     0     0     1    NA     0    NA     2     0     1     0    NA     1
## [34,]     0     0     2     0     0     1    NA    NA     1    NA     0     0
## [35,]     0     0    NA     1    NA     1     0     2     1     0     0    NA
## [36,]    NA     2    NA     0     1    NA     0    NA     2    NA    NA     0
## [37,]     2    NA     1     0     1    NA    NA    NA     0     0     1     0
## [38,]     0     0    NA     1    NA    NA     2     0     0     0     1     1
## [39,]     0     0    NA    NA     1    NA     1     0     1    NA     2     2
## [40,]     1     0    NA    NA     1     1    NA     1     0     0     1    NA
## [41,]     2     0     0    NA    NA    NA     0     1     0     2     0     0
## [42,]    NA     0     0     0     2     0    NA     0     0     0     0     2
## [43,]    NA    NA     0     2     0    NA    NA     0    NA     1     2     0
## [44,]     0     2     0     0     0     1    NA    NA     0     1    NA    NA
## [45,]    NA     0     0    NA     1     0     0     0     0     0    NA     0
## [46,]     1     0     0     0    NA     0     1     1     0     0     0     0
## [47,]     0     2     0     0     1    NA    NA     2     2     1    NA     0
## [48,]     1     0    NA     1    NA    NA     0     0     0    NA     0    NA
## [49,]    NA     2     1     0     0     0     0     0    NA     0     2     0
## [50,]     0    NA     0     0     1    NA    NA     0    NA    NA     1    NA
##       [,38] [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46] [,47] [,48] [,49]
##  [1,]     2     2     2    NA     0     0     1     2     0    NA     2     0
##  [2,]    NA     0    NA     0    NA     0     2     2     0    NA     2    NA
##  [3,]    NA    NA    NA    NA     2     2     0     2     2    NA     2    NA
##  [4,]     2     2     0    NA     2    NA     0     2     0     1     0     0
##  [5,]     2     2    NA    NA     0     1    NA     2     0    NA     2     1
##  [6,]     2     2     0     2     0     2    NA     2     2     1     2     1
##  [7,]     0     0     1     2    NA    NA    NA     0    NA     0     2     1
##  [8,]     1     1     2     2     1     1     2     0    NA    NA     2     0
##  [9,]    NA     0     2     0    NA     2     2     0     2     0    NA    NA
## [10,]    NA     1    NA    NA    NA     1     1     2    NA     2     2     2
## [11,]    NA    NA     0     0    NA     1     1     2     0    NA     0     0
## [12,]     2    NA     0     1     0     0    NA     1     2     0     0    NA
## [13,]    NA    NA    NA    NA     1    NA     2     1     2     0    NA     2
## [14,]    NA    NA     2     1    NA    NA     0     0     0     1     0     0
## [15,]     0     1     0     2     0     0     2     0     0     0    NA     2
## [16,]     0     0     2     2     0    NA    NA    NA    NA     1     0     2
## [17,]     1     2     2     1     0    NA    NA     0     0     2     0     1
## [18,]     2    NA    NA     1     0    NA     0     1     0    NA     0     0
## [19,]     0     0     1     1     1     0     1     2     1     0    NA    NA
## [20,]     0     0    NA     0     1     1     0    NA     2     0     1     2
## [21,]    NA     0     1     0    NA     1     1     1     1     0     0    NA
## [22,]    NA     1    NA     0     0     2    NA     1    NA     1    NA     0
## [23,]     0     2    NA     2    NA    NA     1     0    NA     0     0     1
## [24,]     0     1    NA     1     0     2     1     2     0     1     0     2
## [25,]     1     0    NA    NA     0    NA     1     0     2     0    NA     2
## [26,]     0     2     1    NA    NA    NA     2     0     0     2     1     0
## [27,]     1     1    NA     1     2     1     0    NA     0     0     1     0
## [28,]     1     0     0     0     1     0    NA     0     0     2     1     0
## [29,]    NA    NA     1     0     0     0     1    NA    NA    NA     1     0
## [30,]     0    NA     1     0     1     2     0     1     0     0    NA    NA
## [31,]    NA    NA     2     0    NA    NA    NA    NA    NA    NA    NA    NA
## [32,]    NA     2     0     2     0     0    NA     0    NA     2    NA    NA
## [33,]    NA     0     2     0     0     2    NA    NA     0     2     2     0
## [34,]     0     2    NA     2     1    NA    NA     0    NA    NA     0    NA
```

```
## [35,]     0    NA    NA     2     1    NA     0    NA    NA     0     1    NA
## [36,]    NA     2     0    NA     0     0     0     0     2    NA     0     2
## [37,]     0    NA    NA    NA     2     2     2     1    NA    NA     0     1
## [38,]     0     2     0    NA     0     1    NA     2     0    NA    NA     1
## [39,]     0     0    NA    NA    NA     1     0     0     2     0     2    NA
## [40,]     2    NA     1     2    NA    NA     0     0    NA     0    NA    NA
## [41,]    NA     0     0     0     0    NA    NA     2    NA     0    NA     2
## [42,]    NA    NA    NA     0     0     1    NA     0     0    NA     0    NA
## [43,]    NA     0    NA     0    NA     2    NA     0     0     0     0    NA
## [44,]    NA    NA     0     1    NA     0     1     0    NA    NA     2     1
## [45,]     2    NA    NA     0    NA     0     2     0     0     1     0     1
## [46,]    NA     1    NA     1     1     0     1     0    NA     2     0     0
## [47,]     1     0    NA    NA     0     0     0    NA     0    NA    NA     0
## [48,]    NA     0    NA     0     1     1    NA    NA    NA     2     1     1
## [49,]     0    NA    NA     0     0    NA     2    NA     0    NA     0     0
## [50,]     0    NA    NA    NA     0     0     0     1     0     1     0     2
##       [,50]
##  [1,]    NA
##  [2,]    NA
##  [3,]    NA
##  [4,]     2
##  [5,]     2
##  [6,]    NA
##  [7,]     2
##  [8,]     0
##  [9,]     0
## [10,]     2
## [11,]     0
## [12,]    NA
## [13,]     1
## [14,]     2
## [15,]     0
## [16,]    NA
## [17,]     2
## [18,]     0
## [19,]    NA
## [20,]     1
## [21,]     2
## [22,]     2
## [23,]     1
## [24,]    NA
## [25,]     2
## [26,]     2
## [27,]     0
## [28,]     0
## [29,]     1
## [30,]     1
## [31,]    NA
## [32,]    NA
## [33,]     0
## [34,]     2
## [35,]     0
## [36,]     0
## [37,]     1
```

```
## [38,]     0
## [39,]     1
## [40,]     0
## [41,]     0
## [42,]     0
## [43,]     1
## [44,]     2
## [45,]     2
## [46,]     1
## [47,]     1
## [48,]     2
## [49,]     2
## [50,]     1
```

- We will now learn the `apply` function. This is a handy function that saves writing for loops which should be eschewed in R. Use the apply function to compute a vector whose entries are the standard deviation of each row. Use the apply function to compute a vector whose entries are the standard deviation of each column. Be careful about the NA's! This should be one line.

```
#?apply
apply(R,MARGIN=1,sd,na.rm=TRUE)
```

```
##  [1] 0.78030184399 0.86211556258 0.83212797981 0.84806793314 0.84492824744
##  [6] 0.73611950197 0.88334762710 0.80752760964 0.87141166435 0.82836355919
## [11] 0.79599839534 0.86645874152 0.79681907289 0.88730016753 0.82060166754
## [16] 0.82329180024 0.83359407964 0.88975652100 0.78000215471 0.62389687596
## [21] 0.89348717267 0.72351284664 0.77390598995 0.90563130866 0.80752760964
## [26] 0.77408420033 0.84723257155 0.78537043660 0.85391256383 0.89087080637
## [31] 0.80985828720 0.86602540378 0.78857386432 0.73106345929 0.90433056190
## [36] 0.79176634141 0.85697307545 0.73111868704 0.89746506806 0.77237351493
## [41] 0.86010752016 0.81211855162 0.89066116505 0.64668978758 0.75503361354
## [46] 0.89788727042 0.89322419491 0.85297368222 0.79471941424 0.85723303999
```

```
apply(R,MARGIN=2,sd,na.rm=TRUE)
```

```
##  [1] 0.85215816722 0.82182530102 0.92728015446 0.83591400764 0.80070533423
##  [6] 0.85215816722 0.82043785845 0.79516676548 0.80127738926 0.84540801671
## [11] 0.87735277919 0.83219007599 0.87480946920 0.84983658560 0.74747048183
## [16] 0.93338744432 0.81167944991 0.77390598995 0.80771679264 0.67202150503
## [21] 0.66901468232 0.70647628014 0.80229046222 0.75134288380 0.85208592300
## [26] 0.87038827978 0.89130527156 0.85700278968 0.81211855162 0.74293796182
## [31] 0.77757017987 0.93945503223 0.80064076903 0.83286086471 0.79415478226
## [36] 0.78747706006 0.87010632698 0.86834497091 0.89294371875 0.86380197161
## [41] 0.85588532090 0.70173853734 0.82139397188 0.81996858772 0.88288571145
## [46] 0.89928422716 0.83937205966 0.88616323851 0.84515425473 0.86194473022
```

- Use the `apply` function to compute a vector whose entries are the count of entries that are 1 or 2 in each column. This should be one line.

```
apply(R, MARGIN = 2, function(x){length(x[(x == 1 | x == 2) & !is.na(x)])})
```

```
##  [1] 19 17 21 16 12 19 18 14 17 17 17 17 19 24 23 18 14 16 21 10 11 14 19 15 17
## [26] 18 18 19 15 19 14 16 22 21 12 23 23 14 18 15 19 14 20 20 20 11 16 18 20 26
```

- Use the `split` function to create a list whose keys are the column number and values are the vector of the columns. Look at the last example in the documentation `?split`.

```
#?split
#?list
R_split=split(R,col(R))
R_split[4]
```

```
## $`4`
##  [1]  0 NA  0  2 NA  2  2  0  2  0  1  0 NA NA NA  1  1 NA  1  1  0  0  0  0 NA
## [26]  2  0 NA  0  1  0  2 NA  1 NA  0 NA  2 NA  0  0 NA  0  2 NA  0 NA  0 NA  1
```

- In one statement, use the `lapply` function to create a list whose keys are the column number and values are themselves a list with keys: "min" whose value is the minimum of the column, "max" whose value is the maximum of the column, "pct_missing" is the proportion of missingness in the column and "first_NA" whose value is the row number of the first time the NA appears.

```
#?lapply
length(R_split)
```

```
## [1] 50
```

```
#lapply(R_split,)

#max in each column
max(R_split[[1]],na.rm=TRUE)
```

```
## [1] 2
```

```
#min in each column
min(R_split[[2]],na.rm=TRUE)
```

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

```
#proportion of NA's in each column
apply(R, MARGIN = 2, function(x){length(x[is.na(x)])})/50
```

```
##  [1] 0.24 0.28 0.34 0.32 0.36 0.24 0.38 0.24 0.34 0.26 0.30 0.36 0.24 0.26 0.34
## [16] 0.34 0.30 0.32 0.32 0.38 0.36 0.32 0.24 0.36 0.30 0.34 0.16 0.30 0.32 0.22
## [31] 0.48 0.34 0.22 0.22 0.34 0.26 0.22 0.40 0.34 0.48 0.28 0.30 0.32 0.34 0.20
## [46] 0.34 0.34 0.26 0.30 0.20
```

```
#first_NA row number of the first time the NA appears

#Position(function(x){R_split[x] == NA},x)
#?Position
R_split[4]
```

```
## $`4`
##  [1]  0 NA  0  2 NA  2  2  0  2  0  1  0 NA NA NA  1  1 NA  1  1  0  0  0  0 NA
## [26]  2  0 NA  0  1  0  2 NA  1 NA  0 NA  2 NA  0  0 NA  0  2 NA  0 NA  0 NA  1
```

```
#lapply(R_split,(function(x){}()))
max(R,na.rm=TRUE)
```

```
## [1] 2
```

```
#?max
```

- Set a seed and then create a vector `v` consisting of a sample of 1,000 iid normal realizations with mean -10 and variance 100.

```
set.seed(123)
n=1000
v=rnorm(n,-10,100)
```

- Repeat this exercise by resetting the seed to ensure you obtain the same results.

```
set.seed(123)
v2=rnorm(n,-10,100)
```

- Find the average of `v` and the standard error of `v`.

```
mean(v)
```

```
## [1] -8.3872134065
```

```
#?SE
#?mean
sd(v)/sqrt(length(v))
```

```
## [1] 3.1360148696
```

```
summary(v)
```

```
##         Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
## -290.9774679  -72.8324243   -9.0790361   -8.3872134   56.4601867  314.1039935
```

- Find the 5%ile of `v` and use the `qnorm` function to compute what it theoretically should be. Is the estimate about what is expected by theory?

```
v3=sort(v)
v3[50]
```

```
## [1] -173.63792681
```

```
#?qnorm
qnorm(0.05,-10,100)
```

```
## [1] -174.4853627
```

```
diff=(v3[50])-(qnorm(0.05,-10,100))
diff
```

```
## [1] 0.84743588924
```

```
#Yes, the estimate is close to what is expected by theory, as the difference is <1.
```

- What is the percentile of `v` that corresponds to the value 0? What should it be theoretically? Is the estimate about what is expected by theory?

```
#between 535 and 536 is when the values change from negative to positive, so we can say 53.55%
pnorm(0,-10,100)
```

```
## [1] 0.53982783728
```

```
.5355-pnorm(0,-10,100)
```

```
## [1] -0.004327837277
```

```
#we can say that the estimate is close to what is expected by theory, as the difference is <0.005
```

- Create a function `my_reverse` which takes as required input a vector `v` and returns the vector in reverse where the first entry is the last entry, etc. No function calls are allowed inside your function otherwise

that would defeat the purpose of the exercise! (Yes, there is a base R function that does this called `rev`). Use `head` on `v` and `tail` on `my_reverse(v)` to verify it works.

```
rvrsx=c(rep(NA,1000))
l=length(v)
my_reverse=function(x){

  }

l=length(v)
rvrsv=c(rep(NA,l))
for (i in 1:l){
  rvrsv[i]=v[l+1-i]
}
rvrsv
```

```
##    [1]  -34.919067775540  -62.261669719592 -145.110038570333   97.051603677834
##    [5]  -18.997519701566   21.322877199433  -31.330714323307   35.457780905774
##    [9] -117.420660953035   57.576241547425  155.517581574201 -115.859053594840
##   [13]   68.713361432271  -18.208690377572  -86.948492289028  141.239542683245
##   [17]  -61.675944974373  -40.546963990980 -122.460367125824  220.506198204569
##   [21]    4.641717866796  -42.274636182511   -2.012618132595  -89.951395357653
##   [25] -119.849000709012   88.605822106025 -120.148343893874 -159.869825452676
##   [29] -221.120837326053   -4.526347494257  -68.106738056650  -17.753896679120
##   [33] -177.112705864224 -194.847277533767   -3.023376909970   -0.341416591756
##   [37]   -5.326647248200    5.670298737573  219.961936115445   68.417087909434
##   [41]   85.536563973041 -152.110844750449   21.045008054165   12.071129097065
##   [45] -172.185825912421   25.558761162838   13.363361408987    5.615570727533
##   [49] -110.537758159341  -24.126175728251 -131.394447023951  -37.226753441835
##   [53]  190.668069083189  -76.260727570081    0.363800422830  -55.776053324051
##   [57]  -94.458342853595 -178.591645453874  -45.997511794306   94.908662655933
##   [61]  126.813264802863  -31.450451539977  -82.077363182012  -78.681565244270
##   [65]  -67.541764127117  -11.104582980937  -32.132615316688   -9.714395237777
##   [69] -144.224312528345    8.875310932983   33.109892811499  238.799787722146
##   [73]    9.627804547247  -19.548759044499  121.782088371148   56.515987574597
##   [77]   26.501875143304  -13.653722220474  -31.215053208195  -51.401586304230
##   [81]  149.336995138365   31.276949719403 -202.514525183712  134.566224110115
##   [85]   99.660847172140   48.983592295876 -187.997728796762    0.091985901286
##   [89] -128.528881058722  247.544976372339   16.036149073656   20.131365171840
##   [93]  -14.856835275629   72.592290164749  -72.456747361809   98.461700867092
##   [97]  153.905190875156   19.959368473956  -89.131387881947 -111.411417266147
##  [101]   14.353271876030   27.114795975966  -45.245319922796   14.104593250122
##  [105]  -80.100361225327   36.059134947810  172.853045813536 -121.641641193354
##  [109]   46.204139009905 -109.050763046656   29.529587001350  -86.460601375357
##  [113]  -70.451323033412 -113.813103968721   35.735733162533   44.128414348235
##  [117]  112.096370834882  -28.520216940099 -107.199421147593   37.703723828470
##  [121]  -23.187491182402  114.642390866165  174.326625457782   69.760065758952
##  [125]  185.072100640115  138.377948873809   80.351642636168 -119.955094170275
##  [129]    4.362323466275 -123.920063997340  156.599089998742  -62.723425723392
##  [133]  103.105465184602 -196.022757421841  245.302611346059  -26.328493222929
##  [137]  -96.603774467211 -148.948352398816    4.245843204579   81.120967677132
##  [141]  152.362120768023    4.670848414500  -45.091782719307 -155.884941443809
##  [145]  -70.353124778834  222.086022399448 -125.616739425985   91.645521770216
##  [149]  -19.741249850263   46.226734540307  -12.969397104984  151.554532320720
##  [153]   94.530566396331  -74.356738702711  -12.601863477400   54.979186715352
```

```
##  [157]   -3.575644336996  -55.839013909239  258.485899922680  -85.939811729888
##  [161]   19.908690333615   -4.378515061716  -91.866977751980 -185.106751939608
##  [165]   18.387890635235  -29.177480910994  141.649060169196   68.945985279018
##  [169]  -69.733009136280   95.432227197535 -154.203464635845 -126.961526041411
##  [173] -195.261682424173  -46.835257512651  -69.189210221281  -36.770642155708
##  [177]   68.981792182050  -18.291994233923   64.229702418652 -132.898617755381
##  [181] -130.844272044307 -102.151604286958  111.458879448256  -29.415240932046
##  [185]   72.150678044862   17.687245886161   10.629404484310  -17.767320061056
##  [189] -166.528176503053 -260.791780219725  -25.829402869994   17.631553213579
##  [193]  206.141584630279  -17.562508077984    4.410466246219   17.627455654334
##  [197]  105.293622555862   75.520220506419  -75.801020794948   25.628334486205
##  [201]   54.751335797759   13.111493441816  -54.593502721216   28.836516281937
##  [205]   71.882813653178  -27.063961823861   23.336997015518   78.674903709606
##  [209] -101.356604816499   29.439484798058  -75.377982476010 -198.632515873255
##  [213] -205.820517465693   69.738050054984  -69.299736565532 -141.822072745909
##  [217] -127.669215774973   -9.730833876988   80.443546444970   71.340037407906
##  [221]  -37.907217078731   37.491171403405  -37.845402496359  -10.406065343634
##  [225]  177.786402105219   17.962786243176 -171.777376538123   35.746207935745
##  [229]  -53.682997691312 -176.792804577932  138.783783187804  -42.189258618280
##  [233]  -32.496127104412  117.657868146766   86.785920994706   26.874205801119
##  [237] -179.210152086908   51.656781747743   51.645571635144   43.979060553627
##  [241]  -29.588924882983 -213.768249444080  -30.145814697150   91.283433584425
##  [245]  -22.039382462249  -28.612069937350   11.395797964226   41.147075469813
##  [249]  -20.971032117624  143.843019889679   27.464356800499  227.473471509539
##  [253]  -58.267682179631  259.171400324612 -231.063311057562   10.738115738531
##  [257]   56.441586369409  228.292669523033   89.026224604423   61.160192248624
##  [261]   91.915706933090  -17.526319764873  -19.668607320671   52.203323587106
##  [265]   42.491427931929  -15.605594618627  129.957618540294 -204.365090132398
##  [269]   78.117880892544  163.863376690625   88.636585989546  -83.285380629713
##  [273]    8.942623751481   46.386739030588   15.024782134833   -9.831631151579
##  [277] -162.931053118042   54.183002765162 -182.830451532074 -114.167329428932
##  [281]   30.329033122070   27.324143416641  -38.734079984069   73.440156775704
##  [285]  102.507189227799  134.146175605504  -18.671413498193  -19.363679391105
##  [289]  -31.284827864653   96.109525326174   64.808116249384   12.632494238106
##  [293] -100.421502562064   18.934402881734   48.998267936731 -146.470945798691
##  [297]    1.884943300456  -79.309461401060 -164.044240529001  -82.821911119129
##  [301] -171.403945661912   32.057418869815  -19.745125010562  -26.242194247942
##  [305]  235.806049186527 -164.777654363757  -88.896321811123  109.160126851033
##  [309]  108.718681065156  -19.143427555014 -217.848926886002  -42.132484216839
##  [313]   38.361753383827  -33.902906815256  -51.458872636367 -114.168885898951
##  [317] -109.683898237800  -82.738352908754   88.211300252264   28.230514254637
##  [321]  -79.330452793067  -95.362369672724 -142.775547560413 -121.989972139762
##  [325] -206.170759888530   -0.873261651176  -65.022374148914   29.784220880034
##  [329]  -39.409532715611  -61.798243016782   91.067796451426   46.828862077707
##  [333]  -25.534534714440   10.556120844997   -6.847399540601 -234.905109033240
##  [337]  -12.839505474906  186.624801545563  128.051452831454   76.364843415775
##  [341]    2.071932928703   66.905229499561   43.848203210264   92.025301216443
##  [345]  -92.817427039680   38.062559987522 -155.082431349063   50.877901173463
##  [349]    0.719040701615 -203.850470145338   88.997160862711   64.081452372590
##  [353]  -43.054470011996  -93.081152733625   79.935405320310  -49.579795028890
##  [357]   52.971174669273    5.735334721616   17.106675675055  -93.418823158438
##  [361]  -59.293742416198   53.612403518215  103.939629170942  -21.630251715861
##  [365] -204.295641358195 -145.383433572260   21.023025606090   43.325936275365
##  [369] -195.557165391804  -77.850314541844  198.671743449315 -148.804904760206
```

```
##  [373] -116.349790638204    7.058808184084   84.116580672417  -97.525547749552
##  [377]    7.819018841802   74.643628557969   21.839026040803 -151.528191976949
##  [381]  144.660915422409 -132.292617936591  -77.548229159207  -55.319782999484
##  [385] -270.169967025765   97.494507727239  -11.600252720797  158.758948294570
##  [389] -131.759998776360  -49.703052032784  -94.906113501671  -39.976249664627
##  [393] -151.202579265831  -75.674292536108  -31.073418069178   69.038534388823
##  [397] -161.606762315110  -13.333034227592  -12.734696641374   97.401226028199
##  [401]  175.991086154316  108.118089143852 -204.151837604062  -22.714860682566
##  [405] -125.591652508367   40.412625459161  -38.584542359138   74.053982696100
##  [409]   36.496799078585 -290.977467896054 -141.651040165169   48.299140568006
##  [413]  -88.351578901305   28.142582945135  -32.622198144241 -113.895644030703
##  [417]   39.865804382992 -128.155922705569  -20.234651352060  -41.641586819699
##  [421]   -7.389977467376 -160.775731753668   -2.783324758524   57.325386335484
##  [425]  127.000399397100  -54.478197833699   22.040231431661  -88.173971153208
##  [429] -155.646591707643   86.126415180968   -2.049800467160  -40.174666386638
##  [433]   48.673533909263   43.298928682648    0.107915139528 -128.620658531375
##  [437]  138.403093155628 -118.758071407693   47.872237461645  -93.384358053500
##  [441]  -72.957874268601  -29.726486907630  -62.692517612560  -15.129788612536
##  [445] -155.897072853692  155.191539162316   14.982471992656  101.071141820280
##  [449]  -88.220184561940   74.490424140807  -65.721548194161  233.022665234451
##  [453]   53.949199790053 -129.862235906875   76.415248622589   76.636613211277
##  [457] -129.736350236541   60.416728392972  -56.869977978209  -37.324810681193
##  [461]  -30.224085511028  -70.855717831622   25.885572309315  106.525338994626
##  [465]   70.091433957306  184.585121773144    2.799296568413 -152.056550469574
##  [469]  -60.633354218924   50.537066893033  -65.845691234813   31.027509649043
##  [473]    3.909785716177  113.667580008161   45.004396121414  101.464854536198
##  [477]  -37.594516842554  -79.842066759439  -20.828009123449   86.726726016791
##  [481]  -28.430886884238 -113.968035270204   21.001698657895  -23.180279304118
##  [485]  -83.479925192128  -71.643584287353   10.559750448339   78.246516439659
##  [489]   11.615254180123  -19.736926751308  -17.921170925859    5.012013118814
##  [493] -217.075107054196  -99.594782275650  -19.514745084929 -160.916653732870
##  [497]   65.106130260632   92.678505607490 -109.369859109871  -70.189284598502
##  [501]   45.215771421856   26.411468735506    6.484086791847    8.184719261983
##  [505] -145.807905937061  -97.307100041746 -118.999187241124  119.408390614558
##  [509]  130.405026771417  -20.578416764975  -48.095673929401  152.688121419468
##  [513]  -67.439455218438   81.477327085563   67.886002978353   61.284232003111
##  [517]    2.131837749078  122.821469603699   21.405766350626   -7.954929147497
##  [521]  -32.768406489553   27.300465580492  -42.614389250535  161.230497731928
##  [525]    5.168050450932  -31.404123998538  -33.935156713812 -142.295111275522
##  [529]  -38.148211503897  187.541905401908  102.500274582817   -3.484606742070
##  [533]  162.426223915645   64.090001127427  111.001051044026 -119.278720910151
##  [537]  -49.068497863898   41.850376588825  -44.847244895561   77.196495407874
##  [541]  -39.515780088358   13.061683063120  -58.498759656232  101.027709663882
##  [545] -276.092279846568   81.717491764100   61.517840711106   33.528894890768
##  [549]   94.662884711155  133.040234118818 -173.637926805902 -215.233698390195
##  [553]   78.330281994870   33.881870071445    6.532102124205  -74.511395685979
##  [557]  -29.988982810143   55.990263814261   35.476926898263 -134.104449680849
##  [561]  113.569346230014 -232.498769648741   98.477508986065   61.270332524428
##  [565] -117.742065311198   -8.074072537463 -110.563625951084   20.003854520580
##  [569]   27.842390363683  -29.051680204290 -153.850664491452  153.356842140831
##  [573]   -8.887081277911  229.745248004976  -55.033862392471  -81.846622131926
##  [577]  117.226677946852    2.631585845889   18.642441962882  167.950290977515
##  [581]  -65.527835131325   43.539884086815   33.028469635999  -19.294101847947
##  [585] -274.314895202898  -51.367651359212  -22.270866097738   55.325767947138
```

```
##  [589]   16.783501533168   39.222857006443   12.729192167294  -63.880915997759
##  [593]   65.640643854548  -44.975423921127 -175.054654342511   57.069596873393
##  [597]  -12.884155292831  -73.474826490891 -126.865142442436  -17.355601913648
##  [601]  -45.454240307398 -136.068287881878   74.573154018930  -87.414492960540
##  [605]  140.390060900454  -67.356027076828  -92.947761162452   78.425082002821
##  [609] -225.764633501528  -74.704563131827   24.610361955361  -53.564546969191
##  [613] -123.558847037434   35.623640317981  127.857013695924   37.613327830263
##  [617] -115.501704260268    7.472639698184    9.023031569246   -6.544893286615
##  [621] -189.028123726406  -11.430741316691    4.447570471070 -109.678074220752
##  [625]  -68.948103851500   41.013254687866   72.537986275924  -56.398724296601
##  [629] -175.104889568819  231.677335378821  -56.103833888435   75.692301089932
##  [633] -104.540883112389   27.816777220851  -30.979317122851   71.765944637409
##  [637]   92.467323481835   17.376649103655   55.119328158767  -30.529925746820
##  [641]  247.145814586664 -256.589819376003  -13.303615927599   31.142992061573
##  [645]   23.390294249923  -99.320757005495    1.663728358271  -52.727928720543
##  [649] -209.274848868858   91.494317274366  -27.990623104754 -107.400958280409
##  [653]  -67.185005789556   -0.067240591217 -185.323735914227  -15.198190618090
##  [657]   -4.398326672504  157.105482886294   85.900537778783   52.418747202065
##  [661]  -20.188325571522   76.577940433449  -30.078101558912 -146.403745208238
##  [665] -215.222282043373   27.738797302393  141.921771138818  -82.160444043602
##  [669] -128.843403514798   91.755863695209   -9.270990968310  123.551761505939
##  [673]   53.296071303145  -16.082195466007   58.374552185071  -62.291237631342
##  [677]   56.082029778980 -143.877428723497  111.810861032581   13.743027249103
##  [681]   81.139129179596  158.443570809411 -135.864862806017  -20.567133400377
##  [685]   60.352390275689  -45.204645658262  -70.150670800678 -182.673039911433
##  [689]   65.677476379596  120.117599220059  193.757401824044  113.667504641657
##  [693]   11.198043337229 -211.421049792072   23.117917295898  -53.715953318040
##  [697] -115.251327933874 -103.853870360689  -85.268896821774  -81.524218722280
##  [701]  114.991457096922   -7.901641364576 -145.090268603071  176.685184470686
##  [705]   -3.329912906982  190.248273029283  -67.746800105956  104.526311038036
##  [709]   95.418102337692  106.838387306909    6.806538388466  -31.905037893348
##  [713]  154.084616597749   12.101946856100    3.403864536846    1.814451104668
##  [717]  -36.565162527822   28.602656834968   63.649596477344 -176.747509758566
##  [721]   -7.532501717386   32.816676497051   -2.544882282627    7.480270016126
##  [725]  -54.655721642722   74.964304563336 -149.527434979947   58.430942941646
##  [729]   86.252796848427   74.501300406744   -1.079277692671  -48.877986407174
##  [733] -185.652739555764  -23.315096432894  144.758105898377  219.307897383109
##  [737]  120.019867766682   -5.284556723847  -58.987045313847  -62.111731755252
##  [741] -163.290200289060  -21.363989550614   53.075411565057   98.079949615152
##  [745]  -18.856511213888  149.850877114583   -0.950335286078  -44.391723412846
##  [749]  -66.187636354978  -47.560287166977   38.545997890488  -76.476943527406
##  [753] -145.984070382139  -20.097488532881  180.236182167893  -27.905159438020
##  [757] -123.730362066575  139.606066984635  -60.219871834286  -88.862197085400
##  [761]  -88.153648705475   22.430434416138   31.898240492446  -28.292538837273
##  [765] -141.701613230524  -67.438868976327  -83.852770473957   11.453882662922
##  [769]  -19.031959396585  185.529396549246 -111.559257860354   78.465049897692
##  [773]  -81.721816157401  -80.459646368007   -4.025006261540  -46.365729709525
##  [777]   60.758835383559  100.984813892972   51.798581716653  -67.397347929799
##  [781] -138.471572231780 -133.627311888329  -82.306596993987  -54.116321690529
##  [785]  157.569693240319 -109.250715039204  -61.606383094478  113.247587848534
##  [789]   14.368742959909    1.924523642758  -15.402812508544  155.090746733669
##  [793]  -69.461726745951  -88.860283785024  -57.624689461558  -51.433994791886
##  [797]   44.319405923209  -36.514505669635  121.241297643351  209.881034888372
##  [801] -128.548008459731  -71.116591668042 -135.127136162494   50.070882367242
```

```
##  [805]  189.721338474797 -141.080153332797  -99.536335797754   -0.541647182643
##  [809]  -42.468591149083   11.444530958160  -59.929201717226   65.405378518452
##  [813]   -1.526270780280  100.992028971364  -29.717589434855  -33.627956894110
##  [817]  -96.551286265337  -44.965038795355  116.318517608949 -116.332613397119
##  [821]  -55.836533271111   33.652347891018   21.048074944314   -6.221160082892
##  [825] -119.599626707466  -84.133609627283  202.845189901618  -13.406725373846
##  [829]   -3.470696647468  -31.538050764169   26.896452738509   41.686204431361
##  [833]  -58.378062570874   53.656967403385   19.822759154072  -51.685758816043
##  [837]  314.103993494240 -136.015524475811 -114.917700666607   95.271146557933
##  [841]  -47.458085776701   87.697338668562  -47.243875610383   46.298953322048
##  [845]  -38.039533517025  -21.945260663066 -110.837660827701   23.220257895012
##  [849]   66.904224100091   68.773884747518 -138.703047603518  200.010894052567
##  [853]   58.791677297583 -156.175558499590  -63.090652217030 -170.153617357459
##  [857] -161.466765378175 -167.214415914549  -36.219748940247   60.178433537471
##  [861] -154.389316097180  180.910356921748   63.994751087733 -156.064007092482
##  [865]  103.133721341418 -215.324722154052  -52.249683233962   -5.876707800706
##  [869]   35.150405307921  134.455085842335  -17.130808612360 -106.185663413013
##  [873]   -2.203915043629   13.538657228486  -75.194990169546  174.386200523221
##  [877]  -35.609219219825  -59.055744370067 -104.747461418480    1.764659710013
##  [881] -112.412879060491  -94.970434603358  -74.070600830538    0.567619414894
##  [885]   20.115336216671   41.940720394346  -15.556196552454 -171.788270828916
##  [889]   50.796432222503  -67.534696260839   81.899660906077  -48.022652028776
##  [893] -176.794193658814  -88.490446945708  -14.502772480892 -105.161856726502
##  [897]  -44.754259939773  -34.669187846237   15.688370915653  -81.040656369930
##  [901] -112.642090030678  -33.570035910048  143.261062618519  208.733299301658
##  [905]  -70.025958714713  126.065244853001  -72.790607603937   13.873173511144
##  [909]   44.839695950807   89.350385596212  104.880761845109  -42.593158553123
##  [913]   33.518149083380   99.683901314935   23.178196391570  -32.048656181875
##  [917]   54.437654851883  -47.066003179241   28.528040112633   -9.423581410011
##  [921]  -23.889136243904    8.130347974915 -132.071771225454  -38.477300705101
##  [925]   92.557136969670  -78.800861646736  -80.920076258239   90.573852446226
##  [929] -240.916887564081  -59.103116605654  195.008468562714   82.226746787974
##  [933]   -4.699577326950   34.820977862943   20.352864140426 -117.179122647558
##  [937] -111.857538310709  -43.320738366942  -60.232345310930   27.963948275988
##  [941]   11.594156874397    2.385424384461   48.461374963607 -164.875280423022
##  [945]  141.647060442954  -32.577098565927  126.860228401446  -14.287045729132
##  [949]  -12.854675534870   15.331851399475  -18.336906647183   67.996511833632
##  [953]  -56.665535362322  -50.288483529908 -122.310858320335  110.796199830499
##  [957]  206.895596533851 -136.539635156826  -30.791727801960  -79.470697892051
##  [961]  -48.047100101238  -40.596266373992  -16.191171057672   45.391765353759
##  [965]   58.864025410009   72.158108163749   77.813348753304   79.512566104502
##  [969]  -39.507148299227   32.646422147681  115.381492106993 -123.813693701195
##  [973]    5.337311783652   73.778704449452 -178.669331074241  -72.503926784926
##  [977]  -82.889122929114 -112.600444830724  -31.797491465830 -116.782370598685
##  [981]  -57.279140772793   60.135590156369 -206.661715662964   39.785047822924
##  [985]  168.691313680308  -65.584113475407    1.068271594512   30.077145059405
##  [989]   25.981382705736  112.408179743946  -54.566197009996  -78.685285189353
##  [993] -136.506123460653   36.091620598920  161.506498688328    2.928773516095
##  [997]   -2.949160857542  145.870831414912  -33.017748948328  -66.047564655221
```

```
what=my_reverse(v)
what
```

```
## NULL
```

```
what
```

```
## NULL
```

- Create a function `flip_matrix` which takes as required input a matrix, an argument `dim_to_rev` that returns the matrix with the rows in reverse order or the columns in reverse order depending on the `dim_to_rev` argument. Let the default be the dimension of the matrix that is greater. If the number of rows

```
#TO-DO
```

- Create a list named `my_list` with keys "A", "B", ... where the entries are arrays of size 1, 2 x 2, 3 x 3 x 3, etc. Fill the array with the numbers 1, 2, 3, etc. Make 8 entries according to this sequence.

```
my_list=list()
my_list$A=1
my_list$B=array(1:4,dim=c(2,2))
my_list$C=array(1:27,dim=c(3,3,3))
my_list$D=array(1:4^4,dim=c(4,4,4,4))
my_list$E=array(1:5^5,dim=c(rep(5,5)))
my_list$F=array(1:6^6,dim=c(rep(6,6)))
my_list$G=array(1:7^7,dim=c(rep(7,7)))
my_list$H=array(1:8^8,dim=c(rep(8,8)))

#my_list
```

Run the following code:

```
lapply(my_list, object.size)
```

```
## $A
## 56 bytes
##
## $B
## 232 bytes
##
## $C
## 352 bytes
##
## $D
## 1248 bytes
##
## $E
## 12744 bytes
##
## $F
## 186864 bytes
##
## $G
## 3294416 bytes
##
## $H
## 67109104 bytes
```

```
#?object.size
```

Use `?object.size` to read about what these functions do. Then explain the output you see above. For the later arrays, does it make sense given the dimensions of the arrays?

From C onwards, it does make sense as the rate of increase is really high. Each upward step in alphabet results in an increase of at least 10 times.

#TO-DO

Packages

Install the package `pacman` using regular base R.

```
#install.packages("pacman")
#package is installed
```

First, install the package `testthat` (a widely accepted testing suite for R) from https://github.com/r-lib/testthat using `pacman`. If you are using Windows, this will be a long install, but you have to go through it for some of the stuff we are doing in class. LINUX (or MAC) is preferred for coding. If you can't get it to work, install this package from CRAN (still using `pacman`), but this is not recommended long term.

```
pacman::p_load(testthat)
#package is installed
```

- Create vector `v` consisting of all numbers from -100 to 100 and test using the second line of code su

```
v = seq(-100, 100)
#expect_equal(v, -100 : 101)
```

If there are any errors, the `expect_equal` function will tell you about them. If there are no errors, then it will be silent.

Yes, there was an error as the two sequences are not equal.

Test the `my_reverse` function using the following code:

```
#expect_equal(my_reverse(v), rev(v))
#expect_equal(my_reverse(c("A", "B", "C")), c("C", "B", "A"))
```

A little about strings

- Use the `strsplit` function and `sample` to put the sentences in the string `lorem` below in random order. You will also need to manipulate the output of `strsplit` which is a list. You may need to learn basic concepts of regular expressions.

```
lorem = "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi posuere varius volutpat. Morbi
#?strsplit
sample(strsplit(lorem," "))
```

```
## [[1]]
##  [1] "Lorem"       "ipsum"       "dolor"       "sit"         "amet,"
##  [6] "consectetur" "adipiscing"  "elit."       "Morbi"       "posuere"
## [11] "varius"      "volutpat."   "Morbi"       "faucibus"    "ligula"
## [16] "id"          "massa"       "ultricies"   "viverra."    "Donec"
## [21] "vehicula"    "sagittis"    "nisi"        "non"         "semper."
## [26] "Donec"       "at"          "tempor"      "erat."       "Integer"
## [31] "dapibus"     "mi"          "lectus,"     "eu"          "posuere"
## [36] "arcu"        "ultricies"   "in."         "Cras"        "suscipit"
## [41] "id"          "nibh"        "lacinia"     "elementum."  "Curabitur"
## [46] "est"         "augue,"      "congue"      "eget"        "quam"
## [51] "in,"         "scelerisque" "semper"      "magna."      "Aenean"
## [56] "nulla"       "ante,"       "iaculis"     "sed"         "vehicula"
## [61] "ac,"         "finibus"     "vel"         "arcu."       "Mauris"
```

```
## [66] "at"          "sodales"     "augue."
```

```
#?sample
```

You have a set of names divided by gender (M / F) and generation (Boomer / GenX / Millenial):

- M / Boomer "Theodore, Bernard, Gene, Herbert, Ray, Tom, Lee, Alfred, Leroy, Eddie"
- M / GenX "Marc, Jamie, Greg, Darryl, Tim, Dean, Jon, Chris, Troy, Jeff"
- M / Millennial "Zachary, Dylan, Christian, Wesley, Seth, Austin, Gabriel, Evan, Casey, Luis"
- F / Boomer "Gloria, Joan, Dorothy, Shirley, Betty, Dianne, Kay, Marjorie, Lorraine, Mildred"
- F / GenX "Tracy, Dawn, Tina, Tammy, Melinda, Tamara, Tracey, Colleen, Sherri, Heidi"
- F / Millennial "Samantha, Alexis, Brittany, Lauren, Taylor, Bethany, Latoya, Candice, Brittney, Cheyenne"

Create a list-within-a-list that will intelligently store this data.

```
#HINT:
#strsplit("Theodore, Bernard, Gene, Herbert, Ray, Tom, Lee, Alfred, Leroy, Eddie", split = ", ")[[1]]

list_within_list = list(
  M = list(
    Boomer = strsplit("Theodore, Bernard, Gene, Herbert, Ray, Tom, Lee, Alfred, Leroy, Eddie", split = "
    GenX = strsplit("Marc, Jamie, Greg, Darryl, Tim, Dean, Jon, Chris, Troy, Jeff", split = ", ")[[1]],
    Millennial = strsplit("Zachary, Dylan, Christian, Wesley, Seth, Austin, Gabriel, Evan, Casey, Luis"
    ),
  F = list(
    Boomer = strsplit("Gloria, Joan, Dorothy, Shirley, Betty, Dianne, Kay, Marjorie, Lorraine, Mildred"
    GenX = strsplit("Tracy, Dawn, Tina, Tammy, Melinda, Tamara, Tracey, Colleen, Sherri, Heidi", split =
    Millennial = strsplit("Samantha, Alexis, Brittany, Lauren, Taylor, Bethany, Latoya, Candice, Brittn
  )
)
list_within_list
```

```
## $M
## $M$Boomer
##  [1] "Theodore" "Bernard"  "Gene"     "Herbert"  "Ray"      "Tom"
##  [7] "Lee"      "Alfred"   "Leroy"    "Eddie"
##
## $M$GenX
##  [1] "Marc"   "Jamie"  "Greg"   "Darryl" "Tim"    "Dean"   "Jon"    "Chris"
##  [9] "Troy"   "Jeff"
##
## $M$Millennial
##  [1] "Zachary"   "Dylan"     "Christian" "Wesley"    "Seth"      "Austin"
##  [7] "Gabriel"   "Evan"      "Casey"     "Luis"
##
##
## $F
## $F$Boomer
##  [1] "Gloria"   "Joan"     "Dorothy"  "Shirley"  "Betty"    "Dianne"
##  [7] "Kay"      "Marjorie" "Lorraine" "Mildred"
##
## $F$GenX
##  [1] "Tracy"   "Dawn"    "Tina"    "Tammy"   "Melinda" "Tamara"  "Tracey"
##  [8] "Colleen" "Sherri"  "Heidi"
##
## $F$Millennial
```

```
##  [1] "Samantha" "Alexis"   "Brittany" "Lauren"   "Taylor"   "Bethany"
##  [7] "Latoya"   "Candice"  "Brittney" "Cheyenne"
```

Now cleanup the namespace by deleting all stored objects and functions:

```
rm(list = ls())
```