Homework 1

Suixin Jiang

9/7/2019

```
## Solution 1
#1
vec_1 <- seq(33, 639, 3)
sum(vec_1)
```

```
## [1] 68208
```

```
#2
vec_2 <- seq(33, 639, length = 100)
sum(vec_2)
```

```
## [1] 33600
```

```
#3
vec_3 <- vec_1[-9]
#4
vec_4 <- c(head(vec_1, 8), tail(vec_1, 53))
#5
vec_5 <- c(vec_1, vec_2)
#6
length(vec_5)
```

```
## [1] 303
```

```
#7
sum(log(vec_5))
```

```
## [1] 1701.939
```

```
## Solution 2
library(magrittr)
```

```
## Warning: package 'magrittr' was built under R version 3.5.3
```

```
V <- replicate(1000, sample(v <- seq(1:10), 10, replace = T)) %>%
  mean() %>%
  exp()
V
```

```
## [1] 265.6023
```

```
## Solution 3
# Prof. Zois, I really don't know how to print the exact value with as_tibble
# function based on my current code.
library(tibble)
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
exer <- function(n){
  M <- NULL
  N <- NULL
  for(N in 1:10){ r<-seq(N,N+19,1)^-1
    M<-rbind(M,r) }
```

```
  m <- function(n){ return(m<-matrix(M[1:n,1:n],nrow=n, ncol=n)) }
  v <- function(n){ return(v<-rbind(m(n),colSums(m(n)))) }
  t <- function(n){ return(as_tibble(which(m(n)>0.12 & m(n)<0.15,arr.ind<-T))) }
  print(v(n))
  return(t(n)) }
exer(5)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 1.0000000 0.5000000 0.3333333 0.2500000 0.2000000
## [2,] 0.5000000 0.3333333 0.2500000 0.2000000 0.1666667
## [3,] 0.3333333 0.2500000 0.2000000 0.1666667 0.1428571
## [4,] 0.2500000 0.2000000 0.1666667 0.1428571 0.1250000
## [5,] 0.2000000 0.1666667 0.1428571 0.1250000 0.1111111
## [6,] 2.2833333 1.4500000 1.0928571 0.8845238 0.7456349
```

```
## # A tibble: 5 x 2
##     row   col
##   <int> <int>
## 1     5     3
## 2     4     4
## 3     5     4
## 4     3     5
## 5     4     5
```

```
exer(10)
```

```
##            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
##  [1,] 1.0000000 0.50000000 0.33333333 0.25000000 0.20000000 0.16666667
##  [2,] 0.5000000 0.33333333 0.25000000 0.20000000 0.16666667 0.14285714
##  [3,] 0.3333333 0.25000000 0.20000000 0.16666667 0.14285714 0.12500000
##  [4,] 0.2500000 0.20000000 0.16666667 0.14285714 0.12500000 0.11111111
##  [5,] 0.2000000 0.16666667 0.14285714 0.12500000 0.11111111 0.10000000
##  [6,] 0.1666667 0.14285714 0.12500000 0.11111111 0.10000000 0.09090909
##  [7,] 0.1428571 0.12500000 0.11111111 0.10000000 0.09090909 0.08333333
##  [8,] 0.1250000 0.11111111 0.10000000 0.09090909 0.08333333 0.07692308
##  [9,] 0.1111111 0.10000000 0.09090909 0.08333333 0.07692308 0.07142857
## [10,] 0.1000000 0.09090909 0.08333333 0.07692308 0.07142857 0.06666667
## [11,] 2.9289683 2.01987734 1.60321068 1.34680042 1.16822899 1.03489566
##             [,7]       [,8]       [,9]      [,10]
##  [1,] 0.14285714 0.12500000 0.11111111 0.10000000
##  [2,] 0.12500000 0.11111111 0.10000000 0.09090909
##  [3,] 0.11111111 0.10000000 0.09090909 0.08333333
##  [4,] 0.10000000 0.09090909 0.08333333 0.07692308
##  [5,] 0.09090909 0.08333333 0.07692308 0.07142857
##  [6,] 0.08333333 0.07692308 0.07142857 0.06666667
##  [7,] 0.07692308 0.07142857 0.06666667 0.06250000
##  [8,] 0.07142857 0.06666667 0.06250000 0.05882353
##  [9,] 0.06666667 0.06250000 0.05882353 0.05555556
## [10,] 0.06250000 0.05882353 0.05555556 0.05263158
## [11,] 0.93072899 0.84669538 0.77725094 0.71877140
```

```
## # A tibble: 15 x 2
##      row   col
##    <int> <int>
##  1     7     1
##  2     8     1
```

```
##  3     6     2
##  4     7     2
##  5     5     3
##  6     6     3
##  7     4     4
##  8     5     4
##  9     3     5
## 10     4     5
## 11     2     6
## 12     3     6
## 13     1     7
## 14     2     7
## 15     1     8
```

```
## Solution 4
fizzbuzz <- function(x) {
  by3 <- x %% 3 == 0
  by5 <- x %% 5 == 0
  if (by3 && by5) {
    return('fizzbuzz')
  } else if (by3) {
    return('fizz')
  } else if (by5) {
    return('buzz')
  } else {
    return(x)
  }
}
sapply(0:9, fizzbuzz)
```

```
##  [1] "fizzbuzz" "1"        "2"        "fizz"     "4"        "buzz"
##  [7] "fizz"     "7"        "8"        "fizz"
```