# Simple R Functions

*Ben Gaudiosi*

*February 2, 2018*

**1.**

(a) Write functions `tmpFn1` and `tmpFn2` such that if `xVec` is the vector $(x_1, x_2, ..., x_n)$, then `tmpFn1(xVec)` returns vector $(x_1, x_2^2, ..., x_n^n)$ and `tmpFn2(xVec)` returns the vector $(x_1, \frac{x_2^2}{2}, ..., \frac{x_n^n}{n})$.

---

Here is `tmpFn1`

```
tmpFn1 <- function(xVec){
  return(xVec^(1:length(xVec)))
}

## simple example
a <- c(2, 5, 3, 8, 2, 4)

b <- tmpFn1(a)
b
```

```
## [1]    2   25   27 4096   32 4096
```

and now `tmpFn2`

```
tmpFn2 <- function(xVec2){

  n = length(xVec2)

  return(xVec2^(1:n)/(1:n))
}



c <- tmpFn2(a)
c
```

```
## [1]    2.0000   12.5000    9.0000 1024.0000    6.4000  682.6667
```

(b) Now write a fuction `tmpFn3` which takes 2 arguments $x$ and $n$ where $x$ is a single number and $n$ is a strictly positive integer. The function should return the value of

$$1 + \frac{x}{1} + \frac{x^2}{2} + \frac{x^3}{3} + ... + \frac{x^n}{n}$$

```
tmpFn3 <-function(x,n){
  series =(x^((1:n)))/((1:n))
  return( 1+ sum(series))
}
tmpFn3(2, 5)
```

```
## [1] 18.06667
```

**2. Write a funciton `tmpFn(xVec)` such that if `xVec` is the vector $x = (x_1, ..., x_n)$ then `tmpFn(xVec)` returns the vector of moving averages:**

$$\frac{x_1 + x_2 + x_3}{3}, \frac{x_2 + x_3 + x_4}{3}, ..., \frac{x_{n-2} + x_{n-1} + x_n}{3}$$

```
tmpFn <- function(xVec) {
  len <- length(xVec)
  series <- (xVec[1:(len-2)] + xVec[2:(len-1)] + xVec[3:len])/3
  return(series)
}
```

Try out your function. `tmpFn`(c(1:5,6:1))

```
tmpFn(c(1:5,6:1))
```

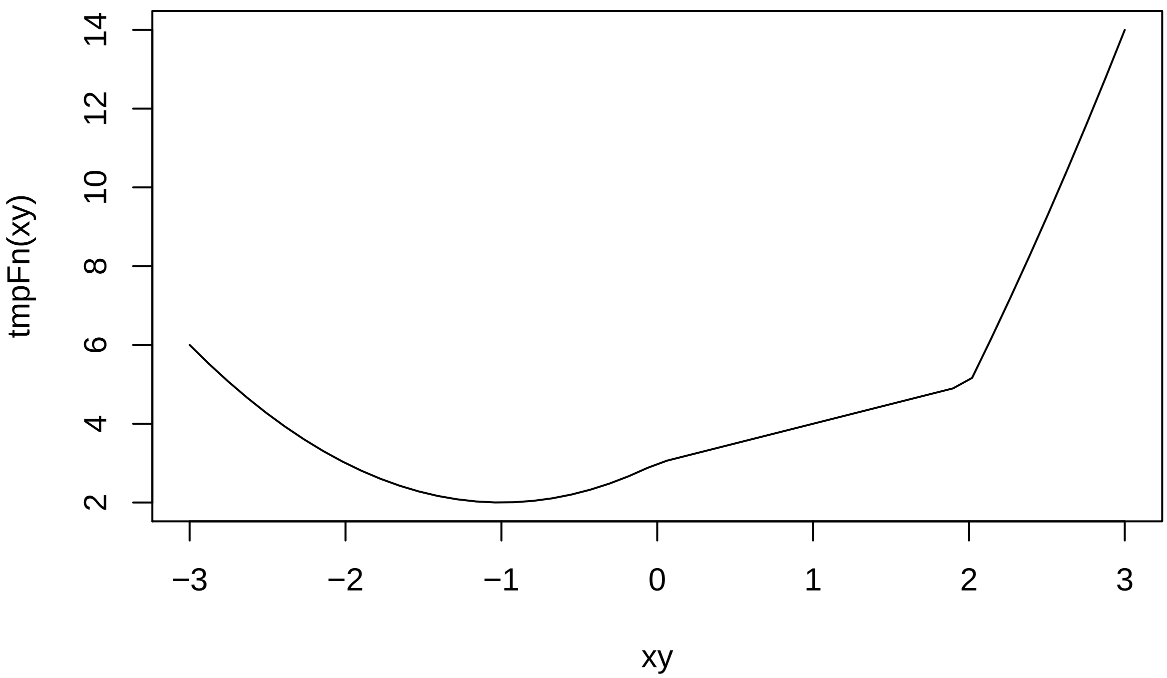
```
## [1] 2.000000 3.000000 4.000000 5.000000 5.333333 5.000000 4.000000 3.000000
## [9] 2.000000
```

**3. Consider the continuous function**

$$f(x) = \begin{cases} x^2 + 2x + 3 & if \quad x < 0 \\ x + 3 & if \quad 0 \leq x < 2 \\ x^2 + 4x - 7 & if \quad 2 \leq x \end{cases}$$

Write a function `tmpFn` which takes a single argument `xVec`. the function should return the vector the values of the function $f(x)$ evaluated at the values in `xVec`.
Hence plot the function $f(x)$ for $-3 < x < 3$.

```
tmpFn <- function(xVec) {
  return(ifelse(xVec < 0, xVec^2  + 2*xVec + 3, ifelse(xVec >= 2, xVec^2 + 4*xVec -7, xVec + 3)))
}
xy <- seq(-3,3,len=50)
plot(xy,tmpFn(xy),type="l")
```




###

4. Write a function which takes a single argument which is a matrix. The function should return a matrix which is the same as the function argument but every odd number is doubled.
Hence the result of using the function on the matrix

$$\begin{bmatrix} 1 & 1 & 3 \\ 5 & 2 & 6 \\ -2 & -1 & -3 \end{bmatrix}$$

should be:

$$\begin{bmatrix} 2 & 2 & 6 \\ 10 & 2 & 6 \\ -2 & -2 & -6 \end{bmatrix}$$

```r
doubleMatrix <- function(matA) {
  return(ifelse(matA %% 2 == 1, 2*matA, matA ))
}
matA <- matrix(c(1,1,3,5,2,6,-2,-1,-3), nrow=3, byrow=TRUE)
matA
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    3
## [2,]    5    2    6
## [3,]   -2   -1   -3
```

```r
doubleMatrix(matA)
```

```
##      [,1] [,2] [,3]
## [1,]    2    2    6
## [2,]   10    2    6
## [3,]   -2   -2   -6
```

**5. Write a function which takes 2 arguements $n$ and $k$ which are positive integers. It should return the $nxn$ matrix:**

$$\begin{bmatrix} k & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & k & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & k & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & k & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & \cdots & k & 1 \\ 0 & 0 & 0 & 0 & \cdots & 1 & k \end{bmatrix}$$

```r
specialMat <- function(n, k) {
  mat <- diag(k, nrow = n)
  mat[abs(col(mat) - row(mat)) == 1] <- 1
  return(mat)
}
specialMat(4, 3)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    1    0    0
## [2,]    1    3    1    0
## [3,]    0    1    3    1
## [4,]    0    0    1    3
```

**6. Suppose an angle $\alpha$ is given as a positive real number of degrees.**

If $0 \leq \alpha < 90$ then it is quadrant 1. If $90 \leq \alpha < 180$ then it is quadrant 2.
if $180 \leq \alpha < 270$ then it is quadrant3. if $270 \leq \alpha < 360$ then it is quadrant 4.
if $360 \leq \alpha < 450$ then it is quadrant 1.
And so on ...

Write a function `quadrant(alpha)` which returns the quadrant of the angle $\alpha$.

```
quadrant <- function(alpha) {
  alpha <- alpha %% 360
  if (alpha < 90) {
    return(1)
  } else if (alpha < 180) {
    return(2)
  } else if (alpha < 270) {
    return(3)
  } else {
    return(4)
  }
}
```

**7.**

(a) Zeller's congruence is the formula:

$$f = ([2.6m - 0.2] + k + y + [y/4] + [c/4] - 2c)mod7$$

where $[x]$ denotes the integer part of $x$; for example $[7.5] = 7$.

Zeller's congruence returns the day of the week $f$ given:

$k$ = the day of the month
$y$ = the year in the century
$c$ = the first 2 digits of the year (the century number)
$m$ = the month number (where January is month 11 of the preceding year, February is month 12 of the preceding year, March is month 1, etc.)
For example, the date 21/07/1'963 has $m = 5, k = 21, c = 19, y = 63$;
the date 21/2/63 has $m = 12, k = 21, c = 19, andy = 62$.
Write a function `weekday(day,month,year)` which returns the day of the week when given the numerical inputs of the day, month and year.
Note that the value of 1 for $f$ denotes Sunday, 2 denotes Monday, etc.

```
zeller <- function(day, month, year) {
  k <- day
  y <- year %% 100
  c <- year %/% 100
  m <- month - 2 + 12*(month < 3)
  result <- (floor(2.6*m - 0.2) + k + y  + floor(y/4) + floor(c/4) - 2*c) %% 7
  return(result)
}
zeller(1,2,2018)
```

```
## [1] 5
```

```r
zeller(2,2,2018)
```

```
## [1] 6
```

(b) Does your function work if the input parameters day, month, and year are vectors with the same length and valid entries?

```r
days <- c(1,1,1,1,1)
months <- c(1,1,1,1,1)
years <- c(2014,2015,2016,2017,2018)
zeller(days,months,years)
```

```
## [1] 4 5 0 1 2
```