# Introduction to R
*Session 2 exercises*

Statistical Consulting Center

1 March, 2017

## 1 Write your own function

(i) In Session 2 you were shown a simple function to calculate the standard error of the mean (SEM), i.e.

```
mystder <- function(x){
     mysd <- sd(x, na.rm = TRUE)
     n <- length(x)
     mysd/sqrt(n)
}
```

Type the above code into your R script and submit it to the R console.

(ii) Modify the function in 1(i) so that the output will have only 2 decimal places.

```r
mystder <- function(x){
       mysd <- sd(x, na.rm = TRUE)
       n <- length(x)
       round(mysd/sqrt(n), 2)
}
```

(iii) Calculate the SEM of `age` using the function you created in 1(ii).

```r
mystder(sports.df$age)

[1] 0.55
```

## 2 Installing an R package

R packages are collections of user-defined functions. The function `std.error`, for example, is contained in the `plotrix` package.

(i) Let's look at what happens when we try to use a function before actually installing on our computer the package in which it is contained. E.g. Calculate the SEM of age using `std.error`.

```
std.error(sports.df$age)

Error in eval(expr, envir, enclos):  could not find function "std.error"
```

(ii) Install the package `plotrix` while in your `R` session by following the instructions below:

(a) Select `Packages` from the bottom right panel of your Rstudio interface.

(b) Click on the `Install Packages` icon just below `Packages`.

(c) Type `plotrix` in the blank space provided below "`Packages (separate multiple with space or comma):`"

(d) Click on `No` if you are asked you to restart `R`

(e) Submit the code `library(plotrix)` to the `R` console to make the functions contained in `plotrix` available in the current `R` session.

```
library(plotrix)
```

(iii) Now, use `std.error` to calculate the standard error of the mean age.

```
std.error(sports.df$age)

[1] 0.5477204
```

(iv) Try writing your own code to calculate the standard error of the mean age. Hint: This only requires one line of code. Use online resources if you cannot remember how the SEM is calculated.

```
with(sports.df, sd(age,na.rm = TRUE)/sqrt(length(age)))

[1] 0.5477204
```

# 3 Subsetting datasets

(i) Produce a one-way frequency table for variable `q1a`.

```
table(sports.df$q1a)



                      Cant choose                              Daily
                               235                                  2
             Several times a month               Several times a week
                                66                                  8
Several times a year or less often
                               649
```

(ii) Which participants chose "Can't choose" for this question?

```
which(sports.df$q1a == "Can?t choose")

integer(0)
```

(iii) Now reproduce the frequency table in 3(i), excluding the participants you identified in 3(ii).

```
excluded <- which(sports.df$q1a == "Can?t choose")
with(sports.df[-excluded, ], table(q1a))

q1a
                      Cant choose                              Daily
                                 0                                  0
             Several times a month               Several times a week
                                 0                                  0
Several times a year or less often
                                 0
```

(iv) Calculate the mean age of male participants.

```
with(sports.df, mean(age[gender == "Male"], na.rm = TRUE))

[1] 52.88503
```

(v) Calculate the mean age of male participants who earn more than $100000 a year.

```
with(sports.df, mean(age[gender == "Male" &
                         income == "> 100 000$"], na.rm = TRUE))

[1] 51
```

(vi) Calculate the mean age of European male participants who earn more than $100000 a year.

```
with(sports.df, mean(age[gender == "Male" &  income == "> 100 000$" &
                         ethnicity == "Europe,White/European"], na.rm = TRUE))

[1] 52.14815
```

## 4 Challenge

Modify the function given in **1**, so that the function will return a 95% confidence interval (with 2 decimal places). Hint: A 95% confidence interval of a variable `x` is given by the mean of `x` ± 1.96 × SEM of `x`. You might find the `paste()` function useful.

```
mystder <- function(x){
       mymean <- mean(x, na.rm = TRUE)
       mysd <- sd(x, na.rm = TRUE)
       n <- length(x)
       mystder = mysd/sqrt(n)
       upperCI = round(mymean + 1.96*mystder, 2)
       lowerCI = round(mymean - 1.96*mystder, 2)
       paste("(", lowerCI, " , ", upperCI, ")", sep = "")
}
mystder(sports.df$age)

[1] "(50.75 , 52.9)"
```