**MAST7210 — Foundations of Data Science**

**Computing Session 5 — Writing Basic Functions in R**

## Task 1 – Summation Notation

**Watch the video "Summation notation" on the moodle page.**

## Task 2 – Examples from the video, use of the for loop

Open RStudio and set your working directory. Start an RScript file to test out the following code that was shown in the video.

### Example 1

Suppose we want to calculate the sum

$$\sum_{x=1}^{4} x = 1 + 2 + 3 + 4 = 10$$

This is equivalent to adding up the integers 1, 2, 3 and 4. In R we can tell it to do:

```
> sum(1:4)
[1] 10
```

### Challenge 1

Suppose we now wanted to calculate

$$\sum_{x=1}^{100} x$$

What R command would we use?

## Example 2

Now suppose we are given the vector n = (10, 20, 30) and we want to calculate:

$$\sum_{i=1}^{3} n_i = n_1 + n_2 + n_3 = 10 + 20 + 30 = 60$$

The first thing we need to do is to define the vector n. We do this with the command:

```
> n <- c(10, 20, 30)
```

Next, we wish to add together the elements of n, we can simply do this using the command:

```
> sum(n)
```

Suppose instead that we wish to create a loop to add each element of n to the previous ones in turn.

We start by defining a dummy variable, sum_n, that initially takes the value 0:

```
> sum_n <- 0
```

We then create a **for loop**. A for loop is used to repeat a task for different input values.

```
> for (i in 1:3) sum_n<-sum_n+n[i]
```

When i=1, sum_n takes the starting value of 0 and then adds on the element n[i]. This then becomes the new value of sum_n. Next when i = 2, n[2] is added onto sum_n, so it is now equal to n[1]+n[2]. Finally when i=3, n[3] is added onto sum_n, so sum_n = n[1]+n[2]+n[3].

## Challenge 2

Suppose we want to calculate the sum of the 2nd 4th and 6th elements ($x_2$, $x_4$ and $x_6$) of a vector x = (2,5,1,8,9,3,4). How would this be coded?

### Example 3

Suppose we wanted to calculate

$$\sum_{x=1}^{4} x(x+1) = (1 \times 2) + (2 \times 3) + (3 \times 4) + (4 \times 5) = 40$$

In R we would code this as:

```
> y<-0
> for (x in 1:4) y<-y+x*(x+1)
> y
[1] 40
```

### Challenge 3

Write code to calculate

$$\sum_{x=1}^{8} x(x+2)$$

## Task 3 – Using the if command

The if statement in R will execute a command if a condition is satisfied. Let us install an R package called MASS which contains a number of interesting data sets. Recall, to install an Rpackage we type:

```
> install.packages("MASS")
```

```
> library("MASS")
```

The data set anorexia contained within this Rpackage contains weight information on anorexia patients pre- and post-treatment.

We first need to attach the anorexia data into our environment. To do this we use the commands:

```
> data(anorexia)
```

```
> attach(anorexia)
```

The attach commands mean we can now call variables Postwt and Prewt directly. If we had not done this we would need to refer to them as anorexia$Postwt and anorexia$Prewt which can be quite cumbersome to type.

Suppose we want to investigate whether patient 1 gained weight after treatment. We could use the command:

```
> if (Postwt[1]>Prewt[1])
{print("Patient 1 gained weight after treatment") } else
{print("Patient did not gain weight after treatment")}
```

Did patient 1 gain weight after treatment?

Repeat the above check for patient number 15. Did patient 15 gain weight after treatment?

## Task 4 – Writing functions in R

Suppose that we wanted to be able to write code to return the sum,

$\sum_{i=1}^{n} i$ = 1+2+3+4+5+…+n

for any value of n you input. In order to do this we need to write a function. A function in R will have the general form (note do not type this into R as it will just produce an error!):

```
func_name <- function(argument) {
statement
}
```

So for the problem above, we can define a function called sum.n:

```
sum.n <- function(n)
{
y<-0
for (i in 1:n) {y<-y+i}
print(y)
}
```

The argument of the function in this case is n, so when we call the function we can tell it what value of the argument we are calculating the function for.

```
> sum.n(5)
[1] 15
> sum.n(10)
[1] 55
```

**Challenge 4**

Write a function called sum.square.n that calculates

$$\sum_{i=1}^{n} i^2$$

Evaluate the function for values of n equal to 4, 6 and 1000.

## Challenge 5

What does the following function do?

```
pow <- function(x, y) {
result <- x^y
print(paste(x,"raised to the power", y, "is", result))
}
```

## Challenge 6

Consider the earlier example of summing only the 2nd, 4th and 6th elements of a defined vector x. Suppose we want to now define a function for this calculation of adding up the elements in the even positions of a vector x, where the length of x might vary. Write code to perform this summation. Test your code using vector x=(4,2,7,8,3,5,8,9,2,4,3,14).

*Hint: The command length(x) will give you the number of elements in vector x*