Class 6: R functions

Kyle Wittkop (A18592410)

Table of contents

Backround

All functions in R have at least 3 things :

- A **name** that we use to call the function
- Has one or more input **arguments**
- The **body**: the lines of R code that do the work

Our first function

Lets write a silly wee function called `add()` to add to add some numbers (the input function)

```
add <- function(x,y) {
  x + y
}
```

Now we can use this function

```
add(100,1)
```

```
[1] 101
```

```
add(x=10,y=10)
```

```
[1] 20
```

```
add( x=c(100,1,100), y=(1))
```

```
[1] 101   2 101
```

```
add(x=c(100,1), y=c(1,100))
```

```
[1] 101 101
```

What if i give a multiple element vector to x and y

```
add(x=c(100,1), y=c(1,100))
```

```
[1] 101 101
```

Q. what if i give three inputs to the function

```
#add(x=c(100,1,y=1,z=1))
```

Q. What if i give only 1 input to the add function?

```
#add(x=100)
addnew <-function(x,y=1){
  x+y
}
```

```
addnew(x=100)
```

```
[1] 101
```

```
addnew(x=100,y=200)
```

```
[1] 300
```

If we write out function with input arguments having no default value than the user will be required to set them when they use the function. we can give our input arguments defualt values by setting them equal to some sensible value e.g. y=1

A second function

Lets make something more interesting: make a sequence generating tool...

The `sample()` function can be a useful starting point

```
sample(1:10, size=4)
```

```
[1] 5 8 7 4
```

Q. Generate 9 random numbers from the input vector from 1 to 10

```
sample(1:10, size=9)
```

```
[1]  4  1  7  9 10  6  2  8  3
```

Q. Generate 12 random numbers from the input vector from 1 to 10?

```
sample(1:10, size =12, replace=TRUE)
```

```
 [1] 2 7 2 8 6 7 4 6 9 7 5 5
```

Q. Write code for the sample function that generates nucleotide sequences of length 6?

```
sample(c("A","T","C","G"), size = 6, replace = TRUE)
```

```
[1] "G" "G" "C" "T" "G" "G"
```

Q. Write a first function `generate_DNA()` that returns a user specified length DNA sequence:

```
generate_DNA <-function(x=6) {
 sample(c("A","T","C","G"), size=x, replace = TRUE)
}
```

```
generate_DNA()
```

```
[1] "A" "A" "A" "T" "C" "C"
```

```
generate_DNA(100)
```

```
  [1] "C" "G" "T" "C" "C" "T" "A" "T" "C" "T" "C" "A" "C" "C" "G" "C" "C" "G"
 [19] "T" "C" "A" "C" "A" "T" "G" "T" "A" "A" "A" "C" "G" "T" "T" "T" "G" "A"
 [37] "G" "T" "C" "T" "T" "A" "G" "G" "T" "C" "T" "G" "C" "T" "C" "C" "C" "C"
 [55] "G" "T" "C" "G" "C" "A" "G" "T" "T" "A" "T" "T" "A" "T" "C" "T" "A" "T"
 [73] "T" "T" "C" "C" "A" "A" "A" "G" "C" "G" "A" "T" "C" "G" "G" "A" "C" "G"
 [91] "A" "G" "T" "C" "A" "G" "A" "G" "A" "A"
```

```
generate_DNA(10)
```

```
 [1] "C" "C" "T" "G" "G" "A" "T" "A" "C" "T"
```

KEY POINTS

Every function in R looks fundamentally the same in terms of its structure, Basically 3 things : name, input, body

```
name <- function(input) {
  Body
}
```

Functions can have multiple inputs. These can be required arguments or "optional" with optional arguments having a set of default values.

> Q. Modify and improve our generate_DNA function to return its generated sequence in a more standard format like "ATGACA" rather than the vector "A","C","G","A"

```
generate_DNA <-function(x=6, fasta=TRUE) {
 ans <- sample(c("A","T","C","G"),
        size=x,  replace = TRUE)
 if(fasta)
   cat("Single-element vector output ")
ans <- paste(ans, collapse="")
  return(ans)
}

generate_DNA(fasta=FALSE)
```

```
[1] "ACACGG"
```

the `paste()` function, its job is to join up or stick together (a.k.a paste) input strings together

```
paste(c("alice","barry"), "love R", sep = " Does not ")
```

```
[1] "alice Does not love R" "barry Does not love R"
```

```
?paste
```

Flow control means where the R brain goes in your code

```
good_mood <- FALSE

if(good_mood) {
  cat("Great!")
} else{
  cat("Bummer")
}
```

```
Bummer
```

A protein generating function

Q. Write a function that generates a user specified length protein sequence.

```
Generate_Prot <-function(x) {
 ans <- sample(c("A","R","N","D","C","Q","E","G","H","I","L","K","M","F","P","S","T","W","Y"
        size=x,  replace = TRUE)
 ans <- paste(ans,collapse="" )
  return(ans)
}
Generate_Prot(6)
```

```
[1] "AQKLIE"
```

Q. Use that function to generate random protien sequences between 6 and 12

```
Generate_Prot(6)
```

```
[1] "AELAYY"
```

```
Generate_Prot(12)
```

```
[1] "CKDGRQMKNTSA"
```

```
for(i in 6:12) {
  cat(">",i, sep="", "\n")
  cat(Generate_Prot(i), "\n")
}
```

```
>6
NLGTDM
>7
LKLNFHH
>8
AVGWHGRN
>9
SNRKLIASV
>10
LMHEGCTKCG
>11
WRIHARIHYLG
>12
LRGLMKNEVQTI
```

Q. are any of your sequences unique i.e not found in nature

Yes, sequences that were 8 amino acids or greater were unique in nature.