

Class 6: R functions

Nicole (PID:A18116280)

All functions in R have at least 3 things:

- A **name**, we pick this and use it to call the function.
- Input **arguments**, there can be multiple comma separated inputs to the function.
- The **body**, lines of R code that do the work of the function.

Our first wee function:

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

Let's test out function

```
add(c(1,2,3), y=10)
```

```
[1] 11 12 13
```

A second function

Let's try something more interesting. Make a sequence generation tool.

The `sample()` function could be useful here.

```
sample(1:10, size = 3)
```

```
[1] 10 1 8
```

change this to work with the nucleotides A C G and T and return 3 of them

```
n <- c("A", "C", "G", "T")
sample(n, size=15, replace = TRUE)
```

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

Turn this snippet into a function that returns a user specified length dna sequence. Let's call it `generate_dna()`...

```
generate_dna <- function(len=10, fasta=FALSE) {
  n <- c("A", "C", "G", "T")
  v <- sample(n, size=len, replace = TRUE)

  #Make a single element vector
  s <- paste(v, collapse="")

  cat("Well done you!\n")

  if(fasta) {
    return(s)
  } else {
    return(v)
  }
}
```

```
generate_dna(5)
```

Well done you!

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

```
s <- generate_dna(15)
```

Well done you!

```
s
```

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

I want the option to return a single element character vector with my sequence all together like this: "GGAGTAC"

```
generate_dna(10, fasta=TRUE)
```

Well done you!

```
[1] "AGCGTCAATA"
```

```
generate_dna(10, fasta=FALSE)
```

Well done you!

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

A more advanced example

Make a third function that generates protein sequence of a user specified length and format

```
generate_protein <- function(size=10, fasta=TRUE) {  
  aa <- c("A", "R", "N", "D", "C", "Q", "E", "G", "H", "I", "L", "K", "M", "F", "P", "S", "T",  
    "Y", "V")  
  seq <- sample(aa, size = size, replace =TRUE)  
  
  if(fasta){  
    return(paste(seq, collapse= ""))  
  } else {  
    return(seq)  
  }  
}
```

try this out...

```
generate_protein(10)
```

```
[1] "EWYQIMFWWE"
```

Q. Generate random protein sequences between lengths 5 and 12 amino acids.

```
generate_protein(5)
```

```
[1] "TQKKK"
```

```
generate_protein(6)
```

```
[1] "NNKFHD"
```

One approach is to do this by brute force calling out function for each length 5 to 12.

Another approach is to write a `for()` loop to iterate over the input valued 5 to 12

A very useful third R specific approach is to use the `sapply()` function.

```
seq_lengths <- 6:12
for (i in seq_lengths) {
  cat(">", i, "\n")
  cat(generate_protein(i) )
  cat("\n")
}
```

```
> 6
TFYMGY
> 7
ARFWVHH
> 8
IQSFSTVC
> 9
PDGKAFCQA
> 10
WPKWVLGDLK
> 11
YDFPEGWWVHQ
> 12
KCFGKAWYYLPK
```

```
sapply(6:12, generate_protein)
```

```
[1] "WQYFCP"      "CPAVDWC"      "SDQAGVRW"     "SRVVDAIMT"    "QSHKYQFMFR"
[6] "IAFKQGAVHHC" "NSTENSQLQKMI"
```

Key-Point: Writing functions in R is doable but not the easiest thing. Starting with a working snippet of code and then using LLM tools to improve and generalize your function code is a productive approach.