Class_07

Dan Vu (A17380158)

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 superated inputs to the function
- The **body**, lines of R code that do the work of the function.

Our first wee function:

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

Let's test our function.

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

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

```r
add(10,100)
```

```
[1] 110
```

A second function

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

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

```r
sample(c("A","C","T","G"), size = 3)
```

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

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

```r
n <- c("A", "C", "T", "G")
sample(n, size = 5, replace=TRUE)
```

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

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

```r
generate_dna <- function(len=10) {
n <- c("A", "C", "T", "G")
v <- sample(n, size = len, replace=TRUE)
seq <- paste(v, collapse = "")
cat("Well done you!\n")
return(v)
}
```

```r
generate_dna(5)
```

```
Well done you!
```

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

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

```
Well done you!
```

```r
s
```

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

I want the option to return a single element vector with 1 sequence all together: "GGAG-TAC"

```
generate_dna <- function(len = 10, fasta = FALSE) {
  n <- c("A", "C", "T", "G")
  v <- sample(n, size = len, replace = TRUE)
  s <- paste(v, collapse = "")
  cat("Well done you!\n")
  if (fasta) {
    return(s)
  } else {
    return(v)
  }
}
```

```
generate_dna((5), fasta=FALSE)
```

```
Well done you!
```

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

##A more advanced example

Make a 3rd function that generates protein sequnece of a user specified length and format.

```
generate_protein <- function(len = 10, fasta = FALSE) {
  n <- c("G", "A", "V", "L", "I", "D", "E", "N", "Q", "P", "F", "W", "K", "C", "M", "Y", "R"
  v <- sample(n, size = len, replace = TRUE)
  s <- paste(v, collapse = "")
  if (fasta) {
    return(s)
  } else {
    return(v)
  }
}
```

```
generate_protein(12, fasta=FALSE)
```

```
 [1] "F" "L" "I" "V" "T" "R" "L" "T" "V" "V" "H" "Q"
```

> Q. Please generate random protein sequences between lenths 5 and 12 amino acids.

```r
lengths <- 5:12
seqs <- lapply(lengths, function(x) generate_protein(len = x, fasta = TRUE))
seqs
```

```
[[1]]
[1] "YYCAP"

[[2]]
[1] "FNKTRW"

[[3]]
[1] "YHTHNEN"

[[4]]
[1] "WGAQTEFM"

[[5]]
[1] "AFHCSNAAR"

[[6]]
[1] "HRIMHRIYYH"

[[7]]
[1] "RHWTGIYQILY"

[[8]]
[1] "STDPEVNAMWHG"
```

```r
sapply(5:12, generate_protein, fasta=TRUE)
```

```
[1] "EDWAH"         "QRKANF"        "CTYGQQY"       "CYYRLHGI"      "ITHSPKKWA"
[6] "CHLMCAMSTM"    "IVDYHQIKLKY"   "EHDTPAPKSMVC"
```

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.