Class 6: R functions

Mai Tamura (PID: A18594079)

All functions in R have at least 3 things:

- A **name**, we pick this and use it to call the function.
- Input **argument**, 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 our function

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

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

```
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.

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

```
[1] 2 6 5
```

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

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

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

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

```r
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)
  }
}
```

```r
generate_dna(5, fasta=TRUE)
```

```
Well done you!
```

```
[1] "CTGAC"
```

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

```
Well done you!
```

```r
s
```

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

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

```r
paste(s, collapse = "")
```

```
[1] "TTGTACCTAGTGCAT"
```

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

```
Well done you!
```

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

A more advanced example

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

```r
generate_protein <- function(len = 15, fasta = TRUE){
  aa <- c("A", "R", "N", "D", "C", "Q", "E", "G", "H", "I",
          "L", "K", "M", "F", "P", "S", "T", "W", "Y", "V")

   seq <- sample(aa, size = len, replace = TRUE)

  if(fasta){
    return(paste(seq, collapse = ""))
  } else {
    return(seq)
  }
}
```

Try this out…

```r
generate_protein(10)
```

```
[1] "HHTKAHHNFH"
```

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

```
generate_protein(5)
```

```
[1] "WAVQH"
```

```
generate_protein(6)
```

```
[1] "DKEPCP"
```

```
generate_protein(7)
```

```
[1] "DLPMFIN"
```

```
generate_protein(8)
```

```
[1] "TSAFCDIG"
```

```
generate_protein(9)
```

```
[1] "DYCASVFKR"
```

```
generate_protein(10)
```

```
[1] "RWSWQGVWSN"
```

```
generate_protein(11)
```

```
[1] "YKNYDVTTDSL"
```

```
generate_protein(12)
```

```
[1] "MIMHPVLLINLY"
```

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

Another approach is to write a `for()` loop to itterate our the input valued 5 to 12.

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

```
seq_lengths <- 5:12
for(i in seq_lengths){
  cat(">", i, "\n", seq="")
  cat(generate_protein(i))
  cat("\n")
}
```

```
> 5
 HMPID
> 6
 SYHTYI
> 7
 GFLNVWT
> 8
 LRFHVPFY
> 9
 ETKWHWNGV
> 10
 WHTPAVGSKD
> 11
 ATWSDQPNMGK
> 12
 RVGPKPREMHMV
```

```
sapply(5:12, generate_protein)
```

```
[1] "QMTIR"         "RDDTQS"        "ARGGMFR"       "AYVHSDNA"      "GWREKYCKQ"
[6] "VCNHLSNQQP"    "FLVNSDFWKAQ"   "SILDHKDYWTFE"
```

> **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.