Class06: R functions

Bobbie Morales (A15443382)

Table of contents

All functions in R have at least 3 things:

-A **name**, we pick this and use it to call our function, -Inout **arguments** (there can be multiple) -The **body** lines of R code that do the work

Our first(silly) function

Write a function to add some numbers

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

now we can call this function:

```
add(c(10, 10), 100)
```

```
[1] 110 110
```

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

```
[1] 110
```

A second function

Write a function to generate random nucleotide sequences of a user specified length: the `sample()` function can be helpful here.

```
v <- sample(c("A", "C", "G", "T"), size=50, replace = TRUE)
```

I want the a 1 element long character vector that looks like "CACAGC"

```
paste(v, collapse = "")
```

```
[1] "GCATGTCCTACGTAAGCTACCTTAATTATCTTCAGGTTGAACCGAAAGCG"
```

Turn this into my first wee function

```
generate_dna <- function(size = 50){
v <- sample(c("A", "C", "G", "T"), size = size, replace = TRUE)
paste(v, collapse = "")
}
```

Test it:

```
generate_dna(60)
```

```
[1] "GACCTATTGCCCTCACTGCGTGAGCATTAATTGGGAACGATTACCCCCCTGGCGGCCACA"
```

```
fasta <- TRUE
if(fasta){
  cat("HELLO You!")
} else {
  cat("No you don't")
}
```

```
HELLO You!
```

Add the ability to return a multi-element vector or a single element fasta like vector.

```
generate_fasta <- function(size = 50, fasta= FALSE) {
v <- sample(c("A", "C", "G", "T"), size = size, replace = TRUE)
s<- paste(v, collapse = "")
    if(fasta) {
  return(s)
      } else {
  return(v)
      }
}
```

]

```
generate_fasta(10)
```

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

a protein generating function

```
generate_protein <- function(size = 50, fasta = TRUE) {
  aa <- c("A", "R", "N", "D", "C", "Q", "E", "G", "H", "I",
          "L", "K", "M", "F", "P", "S", "T", "W", "Y", "V")
  v <- sample(aa, size = size, replace = TRUE)
  s <- paste(v, collapse = "")
  if (fasta) {
    return(s)
  } else {
    return(v)
  }
}
```

```
generate_protein(6)
```

```
[1] "VNCEAF"
```

Use our new `generate_protein()` functin to makerandom protein sequences of length 6 to 12 (i.e one length 6, one length7, etc. up to length 12)

one way to do this is brute force.

```r
generate_protein(6)
```

```
[1] "FCDAMQ"
```

```r
generate_protein(7)
```

```
[1] "GCWGCVW"
```

```r
generate_protein(8)
```

```
[1] "VLIFPFGQ"
```

A second way is to use a `for()` loop

```r
lengths <- 6:12
lengths
```

```
[1]  6  7  8  9 10 11 12
```

```r
for(i in lengths){
  cat(">",i,"\n", sep = "")
  aa <-generate_protein(i)
  cat(aa)
  cat("\n")

}
```

```
>6
LMPWTG
>7
ACAKMER
>8
LLYDRWYD
>9
QVYRKNSVV
>10
RQESCCRSGQ
>11
QWCKKQREAAQ
>12
NVCQVRHHQMTR
```

a third, and better, way to solve this is to use the `apply()` family of functions, specifically the `sapply` function in this case

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

```
[1] "WRTWPR"       "DMLFGEG"      "KPQQTTWV"     "DIILWRLDS"    "SFWYHMPIYG"
[6] "LGQEDKHINMK"  "RSKFQVQFSRTE"
```