Class06: R functions

Ruth Barnes: A16747659

2025-01-23

Today we will get more exposure to functions in R. We call functions to do all our work and today we will learn how to write our own.

A First Silly Function

Note that arguments 3 and 5 have default values (because we set y=0 and z=0).

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

Can I just use this: `add(1,1)`? Yes but 'add' must be defined first/the code above must be run first before the one below can work.

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

```
[1] 2
```

```
add(1, c(10,100))
```

```
[1]  11 101
```

Second line is the same as, `add(x=1, y=c(10,100))`.

What about this: `add(100)`? No it would not work because it does not have a y value.

New function, now with y having a default value. So now `add(100)` will now work.

```r
add <- function(x, y=0) {
  x + y
}
```

```r
add(100)
```

```
[1] 100
```

What would happen if we do: `add(100, 10, 1)`? It would not work because there is an extra variable that is not defined. The new code below will now allow `add(100, 10, 1)` to work.

```r
add <- function (x, y=0, z=0) {
  x + y + z
  }
```

```r
add(100, 10, 1)
```

```
[1] 111
```

A Second More Fun Function

Let's write a function that generates random nucleotide sequences.

We can make use of the in-built `sample()` function in R to help us here.

```r
sample(x=1:10, size=11, replace = TRUE)
```

```
 [1]  7  5  6  2  1  3  4  9  8  9  9
```

Q. Can you use `sample()` to generate a random nucleotide sequence of length 5?

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

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

Q. Write a function `generate_dna()` that makes a nucleotide sequence of a user specified length.

Every function in R has at least 3 things:

- a **name** (in our case, "generate_dna()")
- one or more **input arguments** (the length of sequence we want)
- a **body** (that does the work)

```
generate_dna <- function(length=5) {
  bases <- c("A", "C", "G", "T")
  sample(bases, size=length, replace=TRUE)
}
```

```
generate_dna(10)
```

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

```
generate_dna(100)
```

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

Q. Can you write a `generate_protein()` function that returns amino acid sequence of a user requested length?

```
generate_protein <- function(length=5) {
  aa <- bio3d::aa.table$aa1[1:20]
    sample(aa, size=length, replace=TRUE)
}
```

```
generate_protein(10)
```

```
 [1] "V" "I" "K" "I" "Y" "T" "R" "M" "K" "I"
```

I want my output of this function not to be a vector with one amino acid per element but rather a one element single string.

```r
bases <- c("A", "T", "C", "G")
paste(bases, collapse="")
```

```
[1] "ATCG"
```

```r
generate_protein <- function(length=5) {
  aa <- bio3d::aa.table$aa1[1:20]
    s <- sample(aa, size=length, replace=TRUE)
    paste(s, collapse="")
}
```

```r
generate_protein(
)
```

```
[1] "RFNHT"
```

Q. Generate protein sequences from length 6 to 12

```r
generate_protein(length = 6)
```

```
[1] "TQYTIL"
```

```r
generate_protein(length = 7)
```

```
[1] "NPLGTKH"
```

We can use the useful utility function `sapply()` to help us "apply" our function over all the values 6 to 12.

```r
ans <- sapply(6:12, generate_protein)
```

```r
cat( paste(">ID", 6:12, sep="", "\n", ans, "\n") )
```

```
>ID6
WCTRFH
 >ID7
RTLADDI
```

```
 >ID8
SWTNTYFL
 >ID9
FEVHKWMNC
 >ID10
ARDIDNNCSP
 >ID11
DLAYYCCHWKP
 >ID12
FLTPDWVGEDNK
```

Q. Are any of these sequences unique in nature - i.e. never found in nature. We can search "refseq-protein" and look for 100% Ide and 100% coverage matches with BLASTp.

No matches found.