Class06 Quarto Doc

Dora Deng (PID:A17445600)

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

A first silly function

Note that aruments 2 and 3 have default values (because we set y=0 and z=0) so we don't have to supply them when we call our function.

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

Testing out the function:

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

```
[1] 2
```

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

```
[1]  11 101
```

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

```
[1] 102
```

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= 100, replace = TRUE)
```

```
  [1]  9  9  3  3  4  3  3  2  4  1  6  3  2  2  7  2  2  8  7 10 10 10  1  3  6
 [26]  2  1  4 10  9  7  7  1  3  5  6  8 10 10  3  9  2  3 10  4  4  8  1  6  9
 [51]  3  1  2  6  9  7  8  1  8  3  9  2  2  6  6  2  2  8  2  2  4  2  9  4 10
 [76]  9  4  4  3  2  3 10  9  5 10  5  6 10  9  7  5  7  7  9  6  4  7 10  6  9
```

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

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

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

> Q. Write a function `generated_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)

```r
generated_dna <- function(length){
  bases <- c("A", "T", "G", "C")
  sample (bases, size = length, replace = TRUE)
}
generated_dna(9)
```

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

install.packages("bio3d")

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

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

generate_protein(10)
```

```
[1] "GHTAYNHIAS"
```

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

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

```
[1] "ATGC"
```

> Q. Generate protein sequences from length 6 to 12.

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

```
[1] "IFICGR"
```

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)
ans
```

```
[1] "PNLEKV"       "LQMFNEF"      "NHDGHACH"     "KNTWKMDAY"    "LVIGDINYEN"  
[6] "SPNMKAKIVWW"  "KDTTHPEFRHSH"
```

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

```
>ID.6
PNLEKV
 >ID.7
LQMFNEF
 >ID.8
NHDGHACH
 >ID.9
KNTWKMDAY
 >ID.10
LVIGDINYEN
 >ID.11
SPNMKAKIVWW
 >ID.12
KDTTHPEFRHSH
```

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 significant result was found.