

# Class06: Functions Lab

Ethan Ashley (PID: A15939817)

2025-10-17

## Introduction

All functions in R have at least 3 things”

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

## Our first function

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

Now call this function:

```
add(10)
```

```
## [1] 11
```

## A second function

Write a function to generate random nucleotide sequences of a user specified length:

The `sample()` function can be helpful here. It randomly samples from a supplied vector.

```
randSeq <- function(len) {  
  bases = c("A", "C", "G", "T")  
  paste(sample(bases, len, replace = TRUE), collapse="")  
  #paste can concatenate vectors of characters into a single output  
}
```

```
randSeq(50)
```

```
## [1] "ACTTGGTCCTCTAGTCGAACTACAACCTGCATACTTATGTGGACGCTTTGC"
```

## Flow control

```
fasta <- FALSE
if(fasta) {
  cat("Hello you!")
} else {
  cat("No you dont!")
}
```

## No you dont!

Add the ability to return a multi-element vector or a single element fasta like vector from the random DNA sequence generating function.

```
randSeqFormat <- function(len, fasta=TRUE) {
  #if format is set to true, then the vector will be returned in a fasta format
  bases <- c("A", "C", "G", "T")
  seq <- sample(bases, len, replace = TRUE)

  if(fasta) {
    #paste can concatenate vectors of characters into a single output
    paste(seq, collapse="")
  } else {
    seq
  }
}
```

```
seq <- randSeqFormat(48, TRUE)
seq
```

## [1] "CAACCGTTGGGCGGGGTGAGTGAGTAGGGTAACAATGCATGCTGAGGC"

## A Protein Generating Function

```
generateProtein <- function(len, fasta=TRUE){
  amino_acids = c("A", "R", "N", "D", "E", "G", "H", "I", "L", "K", "F", "P", "S", "T", "W", "V", "C",
  seq <- sample(amino_acids, len, replace = TRUE)

  if(fasta) {
    #paste can concatenate vectors of characters into a single output
    return(paste(seq, collapse=""))
  } else {
    return(seq)
  }
}
```

Use this new function to make random protein sequences of length 6 to 12 (i.e., one w/ 6 AA's, one w/ 7 AA's, etc.)

```
#lengths <- 6:12

for(i in 6:12) {
  cat(">", i, "\n", sep="")
  cat(generateProtein(i))
  cat("\n")
}
```

```
## >6
## GRSVCP
## >7
## SPLGTPE
## >8
## KFPHEAER
## >9
## HHCALNTCR
## >10
## LRFNPKHGPE
## >11
## STIWVGWYSCE
## >12
## NCDARVTDRRDG
```

A better way to solve this is to use the `apply()` family of functions, specifically the `sapply()` function.

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

```
## [1] "SECFFK"      "ACVNTKE"     "EDWGTDRA"    "EFNHHEGLY"   "TGKNAVPNHF"
## [6] "EALYEFLGDKV" "SGTECNHSVGAC"
```