R functions

Cienna Santos (PID: A17581026)

Table of contents

1. Function basics

Let's start writing our first silly funciton to add some numbers:

Every R function has 3 things:

- name (we get to pick this)
- input arguments (there can loads of these separated by a comma)
- the body (the R code that does the work)

```
add <- function(x, y=100, z=0){
  x + y + z
}
```

I can just use this function like any other function as long as R knows about it (i.e. run the code chunk)

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

```
[1] 101
```

```
add( x=c(1,2,3,4), y=100)
```

```
[1] 101 102 103 104
```

```
add(1)
```

```
[1] 101
```

Functions can have "required" input arguments and "optional" input arguments. The optional arguments are defined with an equals default value (`y=10`) in the function definition.

```
add(x=1, y=100, z=10)
```

```
[1] 111
```

> Q. Write a function to return a DNA sequence of a user specified length? Call it `generate_dna()`

The `sample()` function can help here

```
students <- c("jeff", "jeremy", "peter")

sample(students, size = 4, replace=TRUE)
```

```
[1] "peter" "peter" "peter" "peter"
```

2. Generate DNA sequence

Now work with `bases` rather than `students`.

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

Now I have a working 'snippet' of code I can use as the body of my first

```
generate_dna()
```

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

I want the ability to return a sequence like "AGTACCTG" i.e. a one element vector where the bases are all together.

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

```
generate_dna()
```

```
[1] "GTTGC"
```

```
generate_dna(together=F)
```

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

3. Generate Protein function

We can get the set of 20 natural amino-acids from the **bio3d** package.

```
aa <- bio3d::aa.table$aa1[1:20]
```

> Q. Write a protein sequence generation a function that will return sequences of a user specified length?

```
  generate_protein <- function(length=5, together=TRUE){
    ## Get the 20 amino-acids as a vector
    sequence <- sample(aa, size = length, replace=TRUE)

    ##Optionally return a single element string
    if (together){
      sequence <- paste(sequence, collapse = "")
    }
    return(sequence)
}
```

```
generate_protein()
```

```
[1] "CDAMH"
```

Q. Generate random protein sequences of length 6 to 12 amino acids.

```
generate_protein(7)
```

```
[1] "FCGLWVP"
```

```
generate_protein(8)
```

```
[1] "TIKEPNVP"
```

```
generate_protein(9)
```

```
[1] "MGMFAEWKP"
```

We can fix this inability to generate multiple sequences by either editing and adding to the function body code (e.g. a for loop) or by using the R **apply** family of utility functions.

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

```
[1] "GEAKHG"       "CDSACHK"      "NWHRTDAL"     "INFMRNWIK"    "WDFGCHVPKY"
[6] "FECKAKHNSHM"  "SPLYRSEKPTDR"
```

It would be cool and useful if I could get FASTA format output.

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

```
[1] "WKWVTN"       "QFHGVNI"      "CCCIATEL"     "ACSMMCWMS"    "AWAMGGMKVW"
[6] "NRGINREGQGK"  "PKLMEVDWWWTD"
```

```
cat(ans, sep="\n")
```

```
WKWVTN
QFHGVNI
CCCIATEL
ACSMMCWMS
AWAMGGMKVW
NRGINREGQGK
PKLMEVDWWWTD
```

I want this to look like

```
>ID.6
IQIYGMT
>ID.7
WGMATTFQCVMT
>ID.8
NFRRSR
```

The functions `paste()` and `cat()` can help us here…

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

```
>ID.6
WKWVTN
>ID.7
QFHGVNI
>ID.8
CCCIATEL
>ID.9
ACSMMCWMS
>ID.10
AWAMGGMKVW
>ID.11
NRGINREGQGK
>ID.12
PKLMEVDWWWTD
```

```
id.line <- paste(">ID.", 6:12, sep="")
seq.line <- paste(id.line, ans, sep="\n")
cat(seq.line, sep="\n", file="myseq.fa")
```

Q. Determine if these sequences can be found in nature or are they unique?

I BLASTP searched my FASTA format sequences against NR and found that length 6, 7, 8 are not unique and can be found in the databases with 100% coverage and 100% identity.

Random sequences of length 9 and above are unique and can't be found in the databases.