R functions

Sion Kang (PID: A17052234)

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 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.

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

Can I just use this

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

```
[1] 2
```

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

```
[1]  11 101
```

```
add(100)
```

```
[1] 100
```

```
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.

```
sample(x=1:10, size=9)
```

```
[1]  2  6  7  1  5 10  3  9  8
```

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

```
 [1] 3 6 8 3 3 3 1 7 6 3 3
```

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

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

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

```
nt <- c("A", "T", "G", "C")
sample(nt, size=5, replace = TRUE)
```

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

> 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 (R code that does the work)

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

```
generate_dna(10)
```

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

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

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

```
 [1] "A" "R" "N" "D" "C" "Q" "E" "G" "H" "I" "L" "K" "M" "F" "P" "S" "T" "W" "Y"
[20] "V"
```

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

```
generate_protein(10)
```

```
 [1] "D" "M" "I" "K" "N" "W" "Q" "R" "G" "V"
```

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

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

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

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

```
generate_protein(10)
```

```
[1] "QWFWIKYWTV"
```

Q. Generate protein sequences fcrom length 6 to 12?

```
generate_protein(6)
```

```
[1] "NQGKNI"
```

```
generate_protein(7)
```

```
[1] "QACTHGN"
```

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

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

```
[1] "NKITGW"       "GATDMSC"      "EHIGLGKM"     "LKGYWTLMD"    "IRHKRCFVFE"
[6] "YWRLHQQHQFE"  "ELVCAVPNAVFQ"
```

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

```
>ID.6
NKITGW
>ID.7
GATDMSC
>ID.8
EHIGLGKM
>ID.9
LKGYWTLMD
>ID.10
IRHKRCFVFE
>ID.11
YWRLHQQHQFE
>ID.12
ELVCAVPNAVFQ
```

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

My ID.6 and ID.7 had 100% ID and 100% coverage matches but all the longer sequences, ID.8-12 are unique in nature (<100% in ID and coverage). As the length of the random sequence gets longer, the possible permutations is too great; the probability of the sequence having 100% ID and 100% coverage to existing sequences in nature is extremely small for sequences of that length.