Lab 6

Trevor Hoang (A16371830)

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.

First Function

Note that arguments 2 and 3 have default values (because we set y=0 and z=0) so we don't have t osupply them when we call out function.

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

Using the function

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

```
[1] 2
```

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

```
[1]   2 101
```

```
add(100,)
```

```
[1] 100
```

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

```
[1] 111
```

Second 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=11, replace=TRUE)
```

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

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

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

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

> 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(x)
{sample(x=c("A","T","C","G"), size=x, replace=T)}
```

```
generate_dna(10)
```

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

```
generate_dna(100)
```

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

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

```
aa =bio3d::aa.table$aa1[1:20]
generate_protein = function(x) {
  sample(x=aa, size=x, replace=T)
}

generate_protein(5)
```

```
[1] "A" "F" "F" "L" "H"
```

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

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

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

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

generate_protein(5)
```

```
[1] "SIEFN"
```

Q. generate protein sequences from length 6 to 12?

```
generate_protein(6)
```

```
[1] "YEITGK"
```

```
generate_protein(7)
```

```
[1] "NHLAYQA"
```

```
generate_protein(8)
```

```
[1] "KPGSTEFF"
```

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] "DPHTYE"       "MAEPRKT"      "NNDYPLDL"     "FKQGHTVKF"    "EWMTWTLHDH"
[6] "PNWVYVYPKEE"  "LIIRCMFMEMAV"
```

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

```
>ID6
DPHTYE
>ID7
MAEPRKT
>ID8
NNDYPLDL
>ID9
FKQGHTVKF
>ID10
EWMTWTLHDH
>ID11
PNWVYVYPKEE
>ID12
LIIRCMFMEMAV
```

> Q Are any of these sequences unique in nature - i.e. enver found in nature. We can search "refseq-protein" and look for 100% Ide and 100% coverage matches with BLASTp.

About half of the proteins had matches with 100% Ide and 100% coverage. Id9, Id10, Id11 and Id12 did not have matches.