class 6

Rachel Galleta(A16859649)

All functions in R have at least 3 things: -A **name**, we pick this and use it to call the function.

-input **arguments**, there can be multiple comma separated inputs to the function. -the **body**, lines of R code that do the work of the function.

our first function:

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

lets test ot function

```
add(c(1,2,3),y=10)
```

```
[1] 11 12 13
```

```
add(10)
```

```
[1] 11
```

```
add(10,10)
```

```
[1] 20
```

A second function

Lets try something more interesting.Make a sequence generation tool.

the 'sample()' fucntion coulb be useful here.

```
sample(1:10,size=3)
```

```
[1] 7 3 4
```

change this to work with nucleotides A,C,G and T return 3 of them.

```
n<- c("A","C","G","T")
sample(n, size=15, replace= TRUE)
```

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

turn this snipet into a fuction that returns a user specified legnth dna seqeunce, lets call it 'generate_dna()'

```
generate_dna <-function(len=10, fasta=FALSE) {
 n<- c("A","C","G","T")
 v<-sample(n, size=len, replace= TRUE)

 # make a single element vector
 s<-paste(v, collapse ="")

 cat("well done you!")
 if(fasta){
   return(s)
 }else {
   return (v)
 }
}
```

```
generate_dna(5)
```

```
well done you!
```

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

```
s<- generate_dna(15)
```

```
well done you!
```

```
s
```

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

I want the option to return a single element charater vector with my seqence all together like this;“GGAGTAG”

```
s
```

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

```
paste(s,collapse = "")
```

```
[1] "CATGATAGGGGGAAT"
```

```
generate_dna(10,fasta=FALSE)
```

```
well done you!
```

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

A more advanced example

make a thrid function that generates protein sequence of a user specified length and format.

```
generate_protein<-function(size =15,  fasta=TRUE){
  aa <- c("A","R","N","D","C","Q","E","G",
          "H","I","L","K","M","F","P","S",
          "T","W","Y","V")
  seq<-sample(aa,size = size,replace=TRUE)

  if(fasta){
    return(paste(seq, collapse=""))
  }else{
    return(seq)
  }
  }
```

try this out...

```
generate_protein(10,fasta=T)
```

```
[1] "MPNFDRKAKG"
```

Q. generate random protein seqeunces between lenghts 5 and 12 amino-acids.

```
generate_protein(5)
```

```
[1] "PNDML"
```

```
generate_protein(6)
```

```
[1] "FFATPP"
```

one approach is to do this by brute force calling our function for each length 5 to 12.

another approach is to write a 'for c' loop to litetire the input valued 5 to 12.

a very useful third R specific approach is to use the 'supply()' function

```
seq_length<-6:12
for (i in seq_length){
  cat(">",i, "\n",sep="")
  cat(generate_protein(i))
  cat("\n")
}
```

```
>6
TNQGAR
>7
GYNMLAY
>8
MVYAKHAW
>9
TARESGIES
>10
PPCYGPLEDN
>11
KHELCDQVRIA
>12
NSEEYDVRANWW
```

```r
sapply(5:12,generate_protein)
```

```
[1] "MTIMC"         "AVFSMQ"        "ASDFICC"       "VWHDMQEE"      "NSLFAEDKV"
[6] "EKNGQLFVPS"    "KRKYGPRCVIG"   "KGACWVFKTMFA"
```

**key point writting funtion with r is doable, but not the easiest thing. Starting with a working snippet of code and then using LLM tool to improve and generalize your function codes is a productive approach.