class6

Meha Thakur (PID: A16020450)

Table of contents

Lets make a silly function to start with

write a function to add some numbers

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

#call the function
add(5,10)
```

```
[1] 15
```

second function

generate random nucleotide sequences with a user specified length, returning a 1 element option:

```
getsequence<- function(length){
  bases<-c("A","C","T","G")
  seq<-sample(bases,length,replace=T) #need to sample with replacement, otherwise the 'optio
  paste(seq,collapse="") #return one string with all letters, returns 1 element
```

```
}

getsequence(20)
```

```
[1] "CCTAGACCTATCCGGCAGCC"
```

```
getsequence(100)
```

```
[1] "TTGCGCTCTCGATTCTGTTTGTGTGACGCTGACGTGAAGCTCATCCGAAGTTAGTTTTTCGTAGATGCTAAGAGTGTTCCGTCGTGAC
```

```
fasta<-F

  if(fasta){
    cat("Hello You!")
  } else{
    cat("No you dont!")
  }
```

```
No you dont!
```

```
getfasta<- function(length,fasta=T){
  bases<-c("A","C","T","G")

  seq<-sample(bases,length,replace=T) #need to sample with replacement, otherwise the 'optio
  nospace<-paste(seq,collapse="")

  if(fasta==T ){
    return(nospace) #return one string with all letters, returns 1 element
 }else{
   seq
 }

}
```

Protein generating function

```
getProtein<- function(length,fasta=T){
  bases<-c("A","R","N","D","C","E","Q","G","H","I","L","K","M","F","P","S","T","W","Y","V")

  seq<-sample(bases,length,replace=T) #need to sample with replacement, otherwise the 'optio
  nospace<-paste(seq,collapse="")

  if(fasta==T ){
    return(nospace) #return one string with all letters, returns 1 element
 }else{
   return(seq)
 }

}
```

Use new generate protein function to make random sequences of length 6 to 12 (i.e. one 6, one 7, etc. up to length 12). One way to do this is "brute force" (doing it again and again to get to your desired number of sequences). This can get tiresome, so we write for loops.

```
lengths<-6:12
for(i in lengths){

  cat(">",i,"\n")
  aa<-getProtein(i)
  cat(aa)
  cat("\n") #return next sequence on next line

}
```

```
> 6
CYECNE
> 7
EAEECWQ
> 8
AVILGQEW
> 9
PVHYYPFSE
> 10
DSTASIGHGG
> 11
FMSVNYDCDAH
> 12
QDKKMVICEDTN
```

Another approach –> sapply function

```
help(sapply)

length<- 6:12

sapply(lengths,getProtein)
```

```
[1] "EWFHTT"       "MAHKVYK"      "FPIVSSWF"     "QEGHSFWWV"    "FWGRNYRKWV"
[6] "FYNLTRTEWSS"  "AVFCMAERFHCP"
```