Class06

Mari Williams (PID: A15858833)

Table of contents

Creating functions

Functions are defined by 3 parts:

Name: Can be whatever you like

Arguments: The inputs you give the function to process

Body: Bulk of the code that runs based on the inputs given

First function

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

You can add options to executable code like this

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

```
[1] 110 110
```

Function 2

Write a function to generate random nucleotide sequences of a user specified length

```
rand_nuc <- function(x){
  nucleotides <- c("A", "G", "T", "C")
  paste(sample(nucleotides, size=x, replace=TRUE), collapse ="")

}
rand_nuc(20)
```

```
[1] "TATCTGTAAAATAACGAGCC"
```

Add the ability to return a multielement vector or single fasta

```
rand_nuc <- function(x, fasta=TRUE){
  nucleotides <- c("A", "G", "T", "C")
  v<- sample(nucleotides, size=x, replace=TRUE)

    if (fasta==TRUE){
    paste(v, collapse="")
  }
  else
  v

}
rand_nuc(20)
```

```
[1] "ACGGAACCATTCCGTTTTCA"
```

```
rand_nuc(20, fasta=FALSE)
```

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

Generate protein

```
rand_protein <- function(x, fasta = TRUE) {
  # 20 standard amino acids
  amino_acids <- c("A", "R", "N", "D", "C",
                   "Q", "E", "G", "H", "I",
                   "L", "K", "M", "F", "P",
                   "S", "T", "W", "Y", "V")

  # sample random amino acids
  seq_vec <- sample(amino_acids, size = x, replace = TRUE)

  # return as FASTA string or vector
  if (fasta) {
    paste(seq_vec, collapse = "")
  } else {
    seq_vec
  }
}

for (i in c(6:8) ){
  print(rand_protein(i))
}
```

```
[1] "RECQQR"
[1] "WFTDYRF"
[1] "EYQIGLVP"
```

```
sapply(6:8, rand_protein)
```

```
[1] "RHVNSA"   "ETLAWEC"  "EGGPHKTC"
```

Q1

Grading function

```
grade <- function(gradesheet){
  #read in gradesheet and makes a new dataframe from it
  gsheet <- read.csv(gradesheet)
  mean_grades <- data.frame(name = gsheet$X, stringsAsFactors = FALSE)
  mean_grades$grade_average <- NA
```

```
  #make a for loop for each student on the gradesheet
  for (i in 1:nrow(gsheet)){
      rowvec <- as.integer(gsheet[i, 2:6]) #gets all the scores as a numeric vector
      rowvec <- sort(rowvec, na.last = FALSE)[-1] #Orders the vector from smallest to largest
      rowvec[is.na(rowvec)] <- 0 #all remaining NAs will be turned into a 0 for grading purp
      mean_grades$grade_average[i] <- mean(rowvec) #adds the mean average of the remaining gr

  }
  mean_grades #returns the new df
}
grade("student_homework.csv")
```

```
         name grade_average
1   student-1         91.75
2   student-2         82.50
3   student-3         84.25
4   student-4         84.25
5   student-5         88.25
6   student-6         89.00
7   student-7         94.00
8   student-8         93.75
9   student-9         87.75
10 student-10         79.00
11 student-11         86.00
12 student-12         91.75
13 student-13         92.25
14 student-14         87.75
15 student-15         78.75
16 student-16         89.50
17 student-17         88.00
18 student-18         94.50
19 student-19         82.75
20 student-20         82.75
```

Q2

```
grades<-grade("student_homework.csv")
sorted_grades <- grades[order(-grades$grade_average), ]
head(sorted_grades)
```

```
         name grade_average
18 student-18         94.50
7   student-7         94.00
8   student-8         93.75
13 student-13         92.25
1   student-1         91.75
12 student-12         91.75
```

Student 18 has the highest average grade after dropping the lowest score.

Q3

```
gsheet <- read.csv("student_homework.csv")
assignment_means <- colMeans(gsheet[, 2:6], na.rm = TRUE)
assignment_means
```

```
     hw1      hw2      hw3      hw4      hw5
89.00000 80.88889 80.80000 89.63158 83.42105
```

HW2 is the lowest average if we drop the NAs.

If we count the NAs as zero…

```
assignment_means <- apply(gsheet[, 2:6], 2, function(col) {
  col[is.na(col)] <- 0
  mean(col)
})
assignment_means
```

```
  hw1   hw2   hw3   hw4   hw5
89.00 72.80 80.80 85.15 79.25
```

The lowest is still hw2!