

# **HuGen2071 book**

Daniel E. Weeks

August 9, 2022

# Table of contents

<b>Preface</b>	<b>3</b>
<b>1 Preparation</b>	<b>4</b>
1.1 Basic programming ideas . . . . .	4
1.1.1 Introduction to Coding . . . . .	4
1.2 R . . . . .	4
1.2.1 PhD Training Workshop: Statistics in R . . . . .	4
1.3 R and RStudio . . . . .	4
1.3.1 R for the Rest of Us . . . . .	4
1.4 GitHub . . . . .	5
1.5 R Markdown . . . . .	5
1.6 Unix . . . . .	5
<b>2 Introduction</b>	<b>6</b>
<b>3 Logistics</b>	<b>7</b>
3.1 GitHub: Set up an account . . . . .	7
3.2 GitHub Classroom . . . . .	7
<b>4 R Basics Group Exercise</b>	<b>8</b>
4.1 Set up the data frame a . . . . .	8
4.2 Exercise 1 . . . . .	8
4.3 Exercise 2 . . . . .	9
4.4 Exercise 3 . . . . .	9
4.5 Exercise 4 . . . . .	10
4.6 Exercise 5 . . . . .	11
4.7 Exercise 6 . . . . .	12
4.8 Exercise 7 . . . . .	12
4.9 Exercise 8 . . . . .	13
<b>5 Summary</b>	<b>16</b>
<b>References</b>	<b>17</b>

# Preface

This is a Quarto book.

To learn more about Quarto books visit <https://quarto.org/docs/books>.

# 1 Preparation

The first part of our HuGen 2071 course aims to teach you R in the context of applied data wrangling in a genetic context. In our experience, if you have never programmed before, it moves kind of fast. As such, it would be useful to review these sources below.

## 1.1 Basic programming ideas

### 1.1.1 Introduction to Coding

This web page and two short videos discusses how computer programming is very similar to writing a recipe - you have to break a complex project down into precise smaller individual steps.

<https://subjectguides.york.ac.uk/coding/introduction>

## 1.2 R

### 1.2.1 PhD Training Workshop: Statistics in R

This online book has a nice introduction to the concepts of programming, RStudio, and R

[https://bookdown.org/animestina/R\\_Manchester/](https://bookdown.org/animestina/R_Manchester/)

See Chapters 1, 2, and 3

## 1.3 R and RStudio

### 1.3.1 R for the Rest of Us

Acquaint or refresh yourself with R and RStudio — including installing them on your computer with this “R for the Rest of Us course” (24 min of videos + exercises):

<https://rfortherestofus.com/courses/getting-started/>

Slides: <https://rfortherestofus.github.io/getting-started/slides/slides.html>

## 1.4 GitHub

To introduce yourself to GitHub:

<https://guides.github.com/introduction/git-handbook/>

<https://guides.github.com/activities/hello-world/>

## 1.5 R Markdown

To introduce yourself or refresh yourself on R Markdown:

<https://rmarkdown.rstudio.com/> (click on Get Started)

## 1.6 Unix

And finally, to introduce yourself or refresh yourself with Unix (well, Linux in this case, but close enough), try Lessons 1–11 here:

<https://www.webminal.org/>

## 2 Introduction

This is a book created from markdown and executable code using Quarto within RStudio.

Book source code: <https://github.com/DanielEWeeks/HuGen2071>

Created by Daniel E. Weeks

Website: <https://www.publichealth.pitt.edu/home/directory/daniel-e-weeks>

## 3 Logistics

### 3.1 GitHub: Set up an account

Please go to <https://github.com> and set up a GitHub account.

Choose your GitHub user name carefully, as you may end up using it later in a professional context.

### 3.2 GitHub Classroom

As GitHub Classroom will be used to distribute course materials and to submit assignments, it would be best if you get git working on your own computer. The easiest way to do this is to install RStudio, R, and git on your computer.

Please follow the detailed instructions in <https://github.com/jfikel/github-classroom-for-students>

In particular, see Step 5 re generating an ssh key so you don't need to login every time.

## 4 R Basics Group Exercise

### 4.1 Set up the data frame a

```
a <- data.frame(n = 1:4)
dim(a)
```

```
[1] 4 1
```


```
a
```

```
  n
1 1
2 2
3 3
4 4
```

### 4.2 Exercise 1

This exercise should help answer this question: ‘In what type of situations would “recycling” be useful?’

Use recycling to insert into the data frame **a** a column named **rowNum1** that contains a 1 in even rows and a 2 in odd rows.

 Expand to see the answer

```
a$rowNum1 <- c(1,2)
a
```

```
  n rowNum1
1 1         1
2 2         2
```



3	3	1
4	4	2

## 4.3 Exercise 2

Use a `for` loop to insert into the data frame `a` a column named `rowNum2` that contains a 1 in even rows and a 2 in odd rows.

💡 Expand to see the answer

```
# Set value that we want to iterate 1, 2, 1, 2, ...
j <- 1
# Initialize rowNum2 to all missing values
a$rowNum2 <- NA
# Start the for loop, looping over the number of rows in a
for (i in c(1:nrow(a))) {
  # Assign value j to row i
  a$rowNum2[i] <- j
  # Increment j
  j <- j + 1
  # If j is greater than 2, set it back to 1
  if (j > 2) {
    j <- 1
  }
}
```

a

	n	rowNum1	rowNum2
1	1	1	1
2	2	2	2
3	3	1	1
4	4	2	2

## 4.4 Exercise 3

Use a `while` loop to insert into the data frame `a` a column named `rowNum3` that contains a 1 in even rows and a 2 in odd rows.

💡 Expand to see the answer

```
a$rowNum3 = NA
i <- 1 #set index
while(i <= nrow(a)){ #set conditions for while loop

  if ((i %% 2)) { #if statement for when "i" is odd
    a$rowNum3[i] <- 1
  }
  else #else statement for when "i" is even
    a$rowNum3[i] <- 2

  i <- i + 1 #counter for "i", increments by 1 with each loop iteration
}
a
```

	n	rowNum1	rowNum2	rowNum3
1	1	1	1	1
2	2	2	2	2
3	3	1	1	1
4	4	2	2	2

## 4.5 Exercise 4

Use a **repeat** loop to insert into the data frame **a** a column named **rowNum4** that contains a 1 in even rows and a 2 in odd rows.

💡 Expand to see the answer

```
a$rowNum4 = NA
i <- 1 #set index
repeat {

  if ((i %% 2)) { #if statement for when "i" is odd
    a$rowNum4[i] <- 1
  }
  else #else statement for when "i" is even
    a$rowNum4[i] <- 2

  i <- i + 1 #counter for "i", increments by 1 with each loop iteration
  if (i > nrow(a)) {
    break
  }
}
a
```

	n	rowNum1	rowNum2	rowNum3	rowNum4
1	1	1	1	1	1
2	2	2	2	2	2
3	3	1	1	1	1
4	4	2	2	2	2

## 4.6 Exercise 5

Use the `rep` command to insert into the data frame `a` a column named `rowNum5` that contains a 1 in even rows and a 2 in odd rows.

💡 Expand to see the answer


```
# This will only work correctly if nrow(a) is even
a$rowNum5 <- rep(c(1,2), nrow(a)/2)
a
```

	n	rowNum1	rowNum2	rowNum3	rowNum4	rowNum5
1	1	1	1	1	1	1
2	2	2	2	2	2	2

3	3	1	1	1	1	1
4	4	2	2	2	2	2

## 4.7 Exercise 6

Use vector addition to construct a vector of length 4 that contains a 1 in even rows and a 2 in odd rows. Then insert this vector into the data frame `a` into a column named `rowNum6`.

 Expand to see the answer

```
r1 <- rep(1, 4)
r2 <- rep(c(0,1), 2)
r1

[1] 1 1 1 1

r2

[1] 0 1 0 1

r1 + r2

[1] 1 2 1 2

a$rowNum6 <- r1 + r2
a
```

	n	rowNum1	rowNum2	rowNum3	rowNum4	rowNum5	rowNum6
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	1	1	1	1	1	1
4	4	2	2	2	2	2	2

## 4.8 Exercise 7

List all even rows of the data frame `a`.

List rows 3 and 4 of the data frame `a`.

💡 Expand to see the answer

```
# All even rows
a[a$rowNum1==2,]
```

	n	rowNum1	rowNum2	rowNum3	rowNum4	rowNum5	rowNum6
2	2	2	2	2	2	2	2
4	4	2	2	2	2	2	2

```
# All odd rows
a[a$rowNum1==1,]
```

	n	rowNum1	rowNum2	rowNum3	rowNum4	rowNum5	rowNum6
1	1	1	1	1	1	1	1
3	3	1	1	1	1	1	1

## 4.9 Exercise 8

This exercise should help answer this question: “When reading a file, will missing data be automatically represented as NA values, or does that need to be coded/manually curated?”

The tab-delimited file in `testdata.txt` contains the following data:

1	1	1
2	2	2
3	NA	99
4	4	4

Your collaborator who gave you these data informed you that in this file 99 stands for a missing value, as does NA.

However if we use the `read.table` command with its default options to read this in, we fail to accomplish the desired task, as 99 is not reading as a missing value:

```
infile <- "data/testdata.txt"
# Adjust the read.table options to read the file correctly as desired.
b <- read.table(infile)
b
```

```

      V1 V2 V3
1     1  1  1
2     2  2  2
3     3 NA 99
4     4  4  4

```

```
str(b)
```

```

'data.frame':  4 obs. of  3 variables:
 $ V1: int  1 2 3 4
 $ V2: int  1 2 NA 4
 $ V3: int  1 2 99 4

```

Use the `read.table` command to read this file in while automatically setting both the 'NA' and the 99 to NA. This can be done by adjusting the various options of the `read.table` command.

#### Tip

Read the help page for the `read.table` command

#### Expand to see the answer

To read this in properly, we have to let 'read.table' know that there is no header and that which values should be mapped to the missing NA value:

```

b <- read.table(infile, header = FALSE, na.strings = c("NA","99"))
b

```

```

      V1 V2 V3
1     1  1  1
2     2  2  2
3     3 NA NA
4     4  4  4

```

```
str(b)
```

```

'data.frame':  4 obs. of  3 variables:
 $ V1: int  1 2 3 4
 $ V2: int  1 2 NA 4
 $ V3: int  1 2 NA 4

```

```
summary(b)
```

	V1	V2	V3
Min.	:1.00	Min. :1.000	Min. :1.000
1st Qu.	:1.75	1st Qu.:1.500	1st Qu.:1.500
Median	:2.50	Median :2.000	Median :2.000
Mean	:2.50	Mean :2.333	Mean :2.333
3rd Qu.	:3.25	3rd Qu.:3.000	3rd Qu.:3.000
Max.	:4.00	Max. :4.000	Max. :4.000
		NA's :1	NA's :1

## 5 Summary

In summary, this book is a work in progress.



## References