Biostat 203B Homework 1

Due Jan 24, 2025 @ 11:59PM

Kiana Mohammadinik and UID 205928003

Table of contents

Display machine information for reproducibility:

```
sessionInfo()
```

```
R version 4.3.0 (2023-04-21)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS 14.4.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  L

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

time zone: America/Los_Angeles
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base
```

```
loaded via a namespace (and not attached):
 [1] compiler_4.3.0  fastmap_1.1.1   cli_3.6.2       tools_4.3.0
 [5] htmltools_0.5.7 rstudioapi_0.14 yaml_2.3.8      rmarkdown_2.29
 [9] knitr_1.45      jsonlite_1.8.8  xfun_0.50       digest_0.6.34
[13] rlang_1.1.3     evaluate_0.23
```

Q1. Git/GitHub

No handwritten homework reports are accepted for this course. We work with Git and GitHub. Efficient and abundant use of Git, e.g., frequent and well-documented commits, is an important criterion for grading your homework.

1. Apply for the Student Developer Pack at GitHub using your UCLA email. You'll get GitHub Pro account for free (unlimited public and private repositories).
2. Create a **private** repository `biostat-203b-2025-winter` and add `Hua-Zhou` and TA team (`Tomoki-Okuno` for Lec 1; `parsajamshidian` and `BowenZhang2001` for Lec 82) as your collaborators with write permission.
3. Top directories of the repository should be `hw1`, `hw2`, … Maintain two branches `main` and `develop`. The `develop` branch will be your main playground, the place where you develop solution (code) to homework problems and write up report. The `main` branch will be your presentation area. Submit your homework files (Quarto file `qmd`, `html` file converted by Quarto, all code and extra data sets to reproduce results) in the `main` branch.
4. After each homework due date, course reader and instructor will check out your `main` branch for grading. Tag each of your homework submissions with tag names `hw1`, `hw2`, … Tagging time will be used as your submission time. That means if you tag your `hw1` submission after deadline, penalty points will be deducted for late submission.
5. After this course, you can make this repository public and use it to demonstrate your skill sets on job market.

Solution: done

Q2. Data ethics training

This exercise (and later in this course) uses the MIMIC-IV data v3.1, a freely accessible critical care database developed by the MIT Lab for Computational Physiology. Follow the instructions at https://mimic.mit.edu/docs/gettingstarted/ to (1) complete the CITI `Data or Specimens Only Research` course and (2) obtain the PhysioNet credential for using the MIMIC-IV data. Display the verification links to your completion report and completion certificate here. **You must complete Q2 before working on the remaining questions.**

(Hint: The CITI training takes a few hours and the PhysioNet credentialing takes a couple days; do not leave it to the last minute.)

Solution: Here is the Completion Report and Completion Certificate of my CITI training.

Q3. Linux Shell Commands

1. Make the MIMIC-IV v3.1 data available at location `~/mimic`. The output of the `ls -l ~/mimic` command should be similar to the below (from my laptop).

```
# content of mimic folder
ls -l ~/mimic/
```

```
total 48
-rw-r--r--@  1 kiananik  staff  15199 Oct 10 17:29 CHANGELOG.txt
-rw-r--r--@  1 kiananik  staff   2518 Oct 10 18:30 LICENSE.txt
-rw-r--r--@  1 kiananik  staff   2884 Oct 11 18:55 SHA256SUMS.txt
drwxr-xr-x@ 24 kiananik  staff    768 Jan 24 22:09 hosp
drwxr-xr-x@ 11 kiananik  staff    352 Jan 21 06:28 icu
```

Refer to the documentation https://physionet.org/content/mimiciv/3.1/ for details of data files. Do **not** put these data files into Git; they are big. Do **not** copy them into your directory. Do **not** decompress the gz data files. These create unnecessary big files and are not big-data-friendly practices. Read from the data folder `~/mimic` directly in following exercises.

Use Bash commands to answer following questions.

Solution: I downloaded the MIMIC IV v3.1 data and it's available under `~/mimic` folder as requested.

2. Display the contents in the folders `hosp` and `icu` using Bash command `ls -l`. Why are these data files distributed as `.csv.gz` files instead of `.csv` (comma separated values) files? Read the page https://mimic.mit.edu/docs/iv/ to understand what's in each folder.

Solution: Here is the content of `hosp` folder

```
ls -l ~/mimic/hosp/
```

```
total 12328176
-rw-r--r--@ 1 kiananik  staff    19928140 Jun 24  2024 admissions.csv.gz
-rw-r--r--@ 1 kiananik  staff      427554 Apr 12  2024 d_hcpcs.csv.gz
-rw-r--r--@ 1 kiananik  staff      876360 Apr 12  2024 d_icd_diagnoses.csv.gz
-rw-r--r--@ 1 kiananik  staff      589186 Apr 12  2024 d_icd_procedures.csv.gz
-rw-r--r--@ 1 kiananik  staff       13169 Oct  3 10:07 d_labitems.csv.gz
-rw-r--r--@ 1 kiananik  staff    33564802 Oct  3 10:07 diagnoses_icd.csv.gz
-rw-r--r--@ 1 kiananik  staff     9743908 Oct  3 10:07 drgcodes.csv.gz
-rw-r--r--@ 1 kiananik  staff   811305629 Apr 12  2024 emar.csv.gz
-rw-r--r--@ 1 kiananik  staff   748158322 Apr 13  2024 emar_detail.csv.gz
-rw-r--r--@ 1 kiananik  staff     2162335 Apr 12  2024 hcpcsevents.csv.gz
-rw-r--r--@ 1 kiananik  staff  2592909134 Oct  3 10:08 labevents.csv.gz
-rw-r--r--@ 1 kiananik  staff   117644075 Oct  3 10:08 microbiologyevents.csv.gz
-rw-r--r--@ 1 kiananik  staff    44069351 Oct  3 10:08 omr.csv.gz
-rw-r--r--@ 1 kiananik  staff     2835586 Apr 12  2024 patients.csv.gz
-rw-r--r--@ 1 kiananik  staff   525708076 Apr 12  2024 pharmacy.csv.gz
-rw-r--r--@ 1 kiananik  staff   666594177 Apr 12  2024 poe.csv.gz
-rw-r--r--@ 1 kiananik  staff    55267894 Apr 12  2024 poe_detail.csv.gz
-rw-r--r--@ 1 kiananik  staff   606298611 Apr 12  2024 prescriptions.csv.gz
-rw-r--r--@ 1 kiananik  staff     7777324 Apr 12  2024 procedures_icd.csv.gz
-rw-r--r--@ 1 kiananik  staff      127330 Apr 12  2024 provider.csv.gz
-rw-r--r--@ 1 kiananik  staff     8569241 Apr 12  2024 services.csv.gz
-rw-r--r--@ 1 kiananik  staff    46185771 Oct  3 10:08 transfers.csv.gz
```

and content of the `icu` folder

```
ls -l ~/mimic/icu/
```

```
total 8506784
-rw-r--r--@ 1 kiananik  staff       41566 Apr 13  2024 caregiver.csv.gz
-rw-r--r--@ 1 kiananik  staff  3502392765 Apr 13  2024 chartevents.csv.gz
-rw-r--r--@ 1 kiananik  staff       58741 Apr 13  2024 d_items.csv.gz
-rw-r--r--@ 1 kiananik  staff    63481196 Apr 13  2024 datetimeevents.csv.gz
-rw-r--r--@ 1 kiananik  staff     3342355 Oct  3 08:36 icustays.csv.gz
-rw-r--r--@ 1 kiananik  staff   311642048 Apr 13  2024 ingredientevents.csv.gz
-rw-r--r--@ 1 kiananik  staff   401088206 Apr 13  2024 inputevents.csv.gz
-rw-r--r--@ 1 kiananik  staff    49307639 Apr 13  2024 outputevents.csv.gz
-rw-r--r--@ 1 kiananik  staff    24096834 Apr 13  2024 procedureevents.csv.gz
```

These data were distributed as `gz` file because `.csv.gz` files effectively compress big data files such as this and reduce storage space needed for them by decreasing the file size.

3. Briefly describe what Bash commands `zcat`, `zless`, `zmore`, and `zgrep` do.

Solution: `zcat` is used to view the contents of a compressed file without having to uncompress the file. `zless` allows us to view the contents of compressed files withouth having to decompress them first. `zmore` uncompresses files and displays them one screenful at a time. `zgrep` command is used to search within compressed files without explicitly decompressing them first.

4. (Looping in Bash) What's the output of the following bash script?

```
for datafile in ~/mimic/hosp/{a,l,pa}*.gz
do
  ls -l $datafile
done
```

```
-rw-r--r--@ 1 kiananik  staff  19928140 Jun 24  2024 /Users/kiananik/mimic/hosp/admissions.cs
-rw-r--r--@ 1 kiananik  staff  2592909134 Oct  3 10:08 /Users/kiananik/mimic/hosp/labevents.c
-rw-r--r--@ 1 kiananik  staff  2835586 Apr 12  2024 /Users/kiananik/mimic/hosp/patients.csv.g
```

Solution: This loop iterates through all files in the ~/mimic/hosp/ directory that start with `a`, `l`, or `pa` and end with `.gz` and list the details such as size and permissions of each file that matches. In this scenario, the output turned out to be the admissions, labevents, and patients files that matched and their details were listed.

Display the number of lines in each data file using a similar loop. (Hint: combine linux commands `zcat <` and `wc -l`.)

Displaying the number of lines in each data file

```
for datafile in ~/mimic/hosp/{a,l,pa}*.gz
do
  echo -n "$datafile: "
  zcat < "$datafile" | wc -l
done
```

```
/Users/kiananik/mimic/hosp/admissions.csv.gz:   546029
/Users/kiananik/mimic/hosp/labevents.csv.gz:  158374765
/Users/kiananik/mimic/hosp/patients.csv.gz:   364628
```

5. Display the first few lines of `admissions.csv.gz`. How many rows are in this data file, excluding the header line? Each `hadm_id` identifies a hospitalization. How many hospitalizations are in this data file? How many unique patients (identified by `subject_id`) are in this data file? Do they match the number of patients listed in the `patients.csv.gz` file? (Hint: combine Linux commands `zcat <`, `head`/`tail`, `awk`, `sort`, `uniq`, `wc`, and so on.)

Solution: Here's the first few lines of `admissions.csv.gz`

```
zcat < ~/mimic/hosp/admissions.csv.gz | head
```

```
subject_id,hadm_id,admittime,dischtime,deathtime,admission_type,admit_provider_id,admission_l
10000032,22595853,2180-05-06 22:23:00,2180-05-07 17:15:00,,URGENT,P49AFC,TRANSFER FROM HOSPIT
10000032,22841357,2180-06-26 18:27:00,2180-06-27 18:49:00,,EW EMER.,P784FA,EMERGENCY ROOM,HOM
10000032,25742920,2180-08-05 23:44:00,2180-08-07 17:50:00,,EW EMER.,P19UTS,EMERGENCY ROOM,HOS
10000032,29079034,2180-07-23 12:35:00,2180-07-25 17:55:00,,EW EMER.,P06OTX,EMERGENCY ROOM,HOM
10000068,25022803,2160-03-03 23:16:00,2160-03-04 06:26:00,,EU OBSERVATION,P39NWO,EMERGENCY RO
10000084,23052089,2160-11-21 01:56:00,2160-11-25 14:52:00,,EW EMER.,P42H7G,WALK-IN/SELF REFER
10000084,29888819,2160-12-28 05:11:00,2160-12-28 16:07:00,,EU OBSERVATION,P35NE4,PHYSICIAN RE
10000108,27250926,2163-09-27 23:17:00,2163-09-28 09:04:00,,EU OBSERVATION,P40JML,EMERGENCY RO
10000117,22927623,2181-11-15 02:05:00,2181-11-15 14:52:00,,EU OBSERVATION,P47EY8,EMERGENCY RO
```

The number of rows in this data file, excluding the header line, is

```
zcat < ~/mimic/hosp/admissions.csv.gz | tail -n +2 | wc -l
```

```
  546028
```

The number of hospitalizations in this data file is

```
zcat < ~/mimic/hosp/admissions.csv.gz |
tail -n +2 |
cut -d, -f2 |
sort |
uniq |
wc -l
```

```
  546028
```

the same as the number of rows in the file.

Peek the first few lines of `patients.csv.gz`:

```
zcat < ~/mimic/hosp/patients.csv.gz | head
```

```
subject_id,gender,anchor_age,anchor_year,anchor_year_group,dod
10000032,F,52,2180,2014 - 2016,2180-09-09
10000048,F,23,2126,2008 - 2010,
10000058,F,33,2168,2020 - 2022,
10000068,F,19,2160,2008 - 2010,
10000084,M,72,2160,2017 - 2019,2161-02-13
10000102,F,27,2136,2008 - 2010,
10000108,M,25,2163,2014 - 2016,
10000115,M,24,2154,2017 - 2019,
10000117,F,48,2174,2008 - 2010,
```

The number of unique patients in this data file is

```
zcat < ~/mimic/hosp/admissions.csv.gz |
tail -n +2 |
awk -F, '{print $1}' |
sort |
uniq |
wc -l
```

```
223452
```

which is less than the number of paients listed in the `patients.csv.gz` file.

```
zcat < ~/mimic/hosp/patients.csv.gz |
tail -n +2 |
awk -F, '{print $1}' |
sort |
uniq |
wc -l
```

```
364627
```

6. What are the possible values taken by each of the variable `admission_type`, `admission_location`, `insurance`, and `ethnicity`? Also report the count for each unique value of these variables in decreasing order. (Hint: combine Linux commands `zcat`, `head/tail`, `awk`, `uniq -c`, `wc`, `sort`, and so on; skip the header line.)

Solution:

Figuring out the column number for each variable

```
zcat < ~/mimic/hosp/admissions.csv.gz |
head -n 1 |
tr ',' '\n' |
nl
```

```
     1  subject_id
     2  hadm_id
     3  admittime
     4  dischtime
     5  deathtime
     6  admission_type
     7  admit_provider_id
     8  admission_location
     9  discharge_location
    10  insurance
    11  language
    12  marital_status
    13  race
    14  edregtime
    15  edouttime
    16  hospital_expire_flag
```

The count for each unique value of `admission_type` in decreasing order

```
zcat < ~/mimic/hosp/admissions.csv.gz |
tail -n +2 |
awk -F, '{print $6}' |
sort |
uniq -c |
sort -nr
```

```
177459 EW EMER.
119456 EU OBSERVATION
84437 OBSERVATION ADMIT
54929 URGENT
42898 SURGICAL SAME DAY ADMISSION
24551 DIRECT OBSERVATION
21973 DIRECT EMER.
13130 ELECTIVE
7195 AMBULATORY OBSERVATION
```

The count for each unique value of `admission_location` in decreasing order

```
zcat < ~/mimic/hosp/admissions.csv.gz |
tail -n +2 |
awk -F, '{print $8}' |
sort |
uniq -c |
sort -nr
```

```
244179 EMERGENCY ROOM
163228 PHYSICIAN REFERRAL
56227 TRANSFER FROM HOSPITAL
42365 WALK-IN/SELF REFERRAL
12965 CLINIC REFERRAL
8518 PROCEDURE SITE
6317 TRANSFER FROM SKILLED NURSING FACILITY
5837 INTERNAL TRANSFER TO OR FROM PSYCH
5734 PACU
 402 INFORMATION NOT AVAILABLE
 255 AMBULATORY SURGERY TRANSFER
   1
```

The count for each unique value of `insurance` in decreasing order

```
zcat < ~/mimic/hosp/admissions.csv.gz |
tail -n +2 |
awk -F, '{print $10}' |
sort |
uniq -c |
sort -nr
```

```
244576 Medicare
173399 Private
104229 Medicaid
14006 Other
9355
 463 No charge
```

The count for each unique value of `ethnicity` in decreasing order

```
zcat < ~/mimic/hosp/admissions.csv.gz |
tail -n +2 |
awk -F, '{print $13}' |
sort |
uniq -c |
sort -nr
```

```
336538 WHITE
75482 BLACK/AFRICAN AMERICAN
19788 OTHER
13972 WHITE - OTHER EUROPEAN
13870 UNKNOWN
10903 HISPANIC/LATINO - PUERTO RICAN
8287 HISPANIC OR LATINO
7809 ASIAN
7644 ASIAN - CHINESE
6597 WHITE - RUSSIAN
6205 BLACK/CAPE VERDEAN
6070 HISPANIC/LATINO - DOMINICAN
3875 BLACK/CARIBBEAN ISLAND
3495 BLACK/AFRICAN
3478 UNABLE TO OBTAIN
2162 PATIENT DECLINED TO ANSWER
2082 PORTUGUESE
1973 ASIAN - SOUTH EAST ASIAN
1886 WHITE - EASTERN EUROPEAN
1858 HISPANIC/LATINO - GUATEMALAN
1661 ASIAN - ASIAN INDIAN
1526 WHITE - BRAZILIAN
1320 HISPANIC/LATINO - SALVADORAN
1247 AMERICAN INDIAN/ALASKA NATIVE
 920 HISPANIC/LATINO - COLUMBIAN
 883 HISPANIC/LATINO - MEXICAN
 774 SOUTH AMERICAN
 725 HISPANIC/LATINO - HONDURAN
 664 ASIAN - KOREAN
 641 HISPANIC/LATINO - CUBAN
 603 HISPANIC/LATINO - CENTRAL AMERICAN
 596 MULTIPLE RACE/ETHNICITY
 494 NATIVE HAWAIIAN OR OTHER PACIFIC ISLANDER
```

7. The `icustays.csv.gz` file contains all the ICU stays during the study period. How

many ICU stays, identified by `stay_id`, are in this data file? How many unique patients, identified by `subject_id`, are in this data file?

Solution: Figuring out the column number for `stay_id` and `subject_id`

```
zcat < ~/mimic/icu/icustays.csv.gz |
head -n 1 |
tr ',' '\n' |
nl
```

```
     1  subject_id
     2  hadm_id
     3  stay_id
     4  first_careunit
     5  last_careunit
     6  intime
     7  outtime
     8  los
```

The number of ICU stays identified by `stay_id`

```
zcat < ~/mimic/icu/icustays.csv.gz |
tail -n +2 |
awk -F, '{print $3}' |
sort |
uniq |
wc -l
```

```
94458
```

The number of unique patients identified by `subject_id`

```
zcat < ~/mimic/icu/icustays.csv.gz |
tail -n +2 |
awk -F, '{print $1}' |
sort |
uniq |
wc -l
```

```
65366
```

8. *To compress, or not to compress. That's the question.* Let's focus on the big data file `labevents.csv.gz`. Compare compressed gz file size to the uncompressed file size. Compare the run times of `zcat < ~/mimic/labevents.csv.gz | wc -l` versus `wc -l labevents.csv`. Discuss the trade off between storage and speed for big data files. (Hint: `gzip -dk < FILENAME.gz > ./FILENAME`. Remember to delete the large `labevents.csv` file after the exercise.)

Solution:

```
ls -lh ~/mimic/hosp/labevents.csv.gz
gzip -dk ~/mimic/hosp/labevents.csv.gz
ls -lh ~/mimic/hosp/labevents.csv
```

```
-rw-r--r--@ 1 kiananik  staff   2.4G Oct  3 10:08 /Users/kiananik/mimic/hosp/labevents.csv.gz
-rw-r--r--  1 kiananik  staff    17G Oct  3 10:08 /Users/kiananik/mimic/hosp/labevents.csv
```

The runtime for `zcat < ~/mimic/labevents.csv.gz | wc -l` is

```
time zcat < ~/mimic/hosp/labevents.csv.gz | wc -l
```

```
 158374765

real    0m18.726s
user    0m28.806s
sys 0m2.004s
```

The runtime for `wc -l labevents.csv`

```
time wc -l ~/mimic/hosp/labevents.csv
```

```
 158374765 /Users/kiananik/mimic/hosp/labevents.csv

real    0m19.731s
user    0m17.318s
sys 0m1.580s
```

Deleting the `labevents.csv` file

```
rm ~/mimic/hosp/labevents.csv
```

The difference between the two runtimes shows that even though compressed files save significant storage space, they also require decompression for usage which can be slower than a regular file that doesn't need decompression.

Q4. Who's popular in Price and Prejudice

1. You and your friend just have finished reading *Pride and Prejudice* by Jane Austen. Among the four main characters in the book, Elizabeth, Jane, Lydia, and Darcy, your friend thinks that Darcy was the most mentioned. You, however, are certain it was Elizabeth. Obtain the full text of the novel from http://www.gutenberg.org/cache/epub/42671/pg42671.txt and save to your local folder.

```
wget -nc http://www.gutenberg.org/cache/epub/42671/pg42671.txt
```

```
File ‘pg42671.txt’ already there; not retrieving.
```

Explain what `wget -nc` does. Do **not** put this text file `pg42671.txt` in Git. Complete the following loop to tabulate the number of times each of the four characters is mentioned using Linux commands.

Solution: `wget -nc` has two parts where `wget` is used to downlaod files from the web and `-nc` which stands for no-clobber prevents the code from overwriting an already existing file by not allowing it to download again if the file already exists.

```
wget -nc http://www.gutenberg.org/cache/epub/42671/pg42671.txt
for char in Elizabeth Jane Lydia Darcy
do
  echo $char:
  # some bash commands here
done
```

```
File ‘pg42671.txt’ already there; not retrieving.

Elizabeth:
Jane:
Lydia:
Darcy:
```

2. What's the difference between the following two commands?

```
echo 'hello, world' > test1.txt
```

and

```
echo 'hello, world' >> test2.txt
```

Solution: the command with > operator overwrites the file test1.txt with the text hello, world while the command with the >> appends the given text hello, world to the file test2.txt. With the >> operator, if the file test2.txt already exists, the text hello, world will be added to the end of the file instead of overwriting it.

3. Using your favorite text editor (e.g., `vi`), type the following and save the file as `middle.sh`:

```
#!/bin/sh
# Select lines from the middle of a file.
# Usage: bash middle.sh filename end_line num_lines
head -n "$2" "$1" | tail -n "$3"
```

Using `chmod` to make the file executable by the owner, and run

```
./middle.sh pg42671.txt 20 5
```

```
Release date: May 9, 2013 [eBook #42671]

Language: English
```

Explain the output. Explain the meaning of `"$1"`, `"$2"`, and `"$3"` in this shell script. Why do we need the first line of the shell script?

Solution: The output shows the date May 9, 2013, the format of the ebook, and it also shows that the language of ht ebook is English. `"$1"` refers to the first argument which is the filename. `"$2"` refers to the second argument which is the number of lines from the start. `"$3"` refers to the third argument which is the number of lines to extract from the end of the previously selected lines. This resulted in the last 5 lines from the first 20 lines in the Pride and Prejudice text to be extracted. We need the first line in the shell script becuase it specifies which interpreter to use when executing the script and without it the script might not run properly.

Q5. More fun with Linux

Try following commands in Bash and interpret the results: `cal`, `cal 2025`, `cal 9 1752` (anything unusual?), `date`, `hostname`, `arch`, `uname -a`, `uptime`, `who am i`, `who`, `w`, `id`, `last | head`, `echo {con,pre}{sent,fer}{s,ed}`, `time sleep 5`, `history | tail`.

Solution:

```
cal
```

```
    January 2025
Su Mo Tu We Th Fr Sa
          1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
```

```
cal 2025
```

```
                               2025

      January               February                March
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
          1  2  3  4                     1                     1
 5  6  7  8  9 10 11   2  3  4  5  6  7  8   2  3  4  5  6  7  8
12 13 14 15 16 17 18   9 10 11 12 13 14 15   9 10 11 12 13 14 15
19 20 21 22 23 24 25  16 17 18 19 20 21 22  16 17 18 19 20 21 22
26 27 28 29 30 31     23 24 25 26 27 28     23 24 25 26 27 28 29
                                            30 31

       April                  May                   June
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
       1  2  3  4  5                  1  2  3   1  2  3  4  5  6  7
 6  7  8  9 10 11 12   4  5  6  7  8  9 10   8  9 10 11 12 13 14
13 14 15 16 17 18 19  11 12 13 14 15 16 17  15 16 17 18 19 20 21
20 21 22 23 24 25 26  18 19 20 21 22 23 24  22 23 24 25 26 27 28
27 28 29 30           25 26 27 28 29 30 31  29 30

        July                 August              September
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
```

```
       1  2  3  4  5                  1  2      1  2  3  4  5  6
 6  7  8  9 10 11 12   3  4  5  6  7  8  9   7  8  9 10 11 12 13
13 14 15 16 17 18 19  10 11 12 13 14 15 16  14 15 16 17 18 19 20
20 21 22 23 24 25 26  17 18 19 20 21 22 23  21 22 23 24 25 26 27
27 28 29 30 31        24 25 26 27 28 29 30  28 29 30
                      31

      October               November              December
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
          1  2  3  4                     1      1  2  3  4  5  6
 5  6  7  8  9 10 11   2  3  4  5  6  7  8   7  8  9 10 11 12 13
12 13 14 15 16 17 18   9 10 11 12 13 14 15  14 15 16 17 18 19 20
19 20 21 22 23 24 25  16 17 18 19 20 21 22  21 22 23 24 25 26 27
26 27 28 29 30 31     23 24 25 26 27 28 29  28 29 30 31
                      30
```

```
cal 9 1752
```

```
   September 1752
Su Mo Tu We Th Fr Sa
       1  2 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

The unusual occurance in the ouput for `cal 9 1752` is that the calender skips sept 3-13 in 1752. Upon further research, I found the reason for this is that 1752 was the year that in the British Empire adopted the Gregorian calendar and 11 days were dropped.

```
date
```

```
Fri Jan 24 22:12:21 PST 2025
```

```
hostname
```

```
Kianas-MBP-2.lan
```

```
arch
```

```
arm64
```

```
uname -a
```

```
Darwin Kianas-MBP-2.lan 23.4.0 Darwin Kernel Version 23.4.0: Fri Mar 15 00:10:42 PDT 2024; r
```

```
uptime
```

```
22:12  up 20 days,  8:08, 1 user, load averages: 3.77 3.53 2.83
```

```
who am i
```

```
kiananik                      Jan 24 22:12
```

```
who
```

```
kiananik         console      Jan 23 15:23
```

```
w
```

```
22:12  up 20 days,  8:08, 1 user, load averages: 3.77 3.53 2.83
USER       TTY      FROM    LOGIN@  IDLE WHAT
kiananik   console  -      Thu15   30:48 -
```

```
id
```

```
uid=501(kiananik) gid=20(staff) groups=20(staff),12(everyone),61(localaccounts),79(_appserve
```

```
last | head
```

```
kiananik   ttys000                        Fri Jan 24 13:36 - 13:36  (00:00)
kiananik   ttys000                        Fri Jan 24 12:51 - 12:51  (00:00)
kiananik   ttys000                        Fri Jan 24 01:45 - 01:45  (00:00)
kiananik   ttys000                        Thu Jan 23 15:23 - 15:23  (00:00)
kiananik   console                        Thu Jan 23 15:23   still logged in
kiananik   ttys000                        Thu Jan 23 15:23 - 15:23  (00:00)
kiananik   ttys000                        Wed Jan 22 01:30 - 01:30  (00:00)
kiananik   ttys001                        Tue Jan 21 15:32 - 15:32  (00:00)
kiananik   ttys001                        Tue Jan 21 15:29 - 15:29  (00:00)
kiananik   ttys000                        Tue Jan 21 15:20 - 15:20  (00:00)
```

```
echo {con,pre}{sent,fer}{s,ed}
```

```
consents consented confers confered presents presented prefers prefered
```

```
time sleep 5
```

```
real    0m5.009s
user    0m0.000s
sys 0m0.001s
```

```
history | tail
```

Done.

Q6. Book

1. Git clone the repository https://github.com/christophergandrud/Rep-Res-Book for the book *Reproducible Research with R and RStudio* to your local machine. Do **not** put this repository within your homework repository `biostat-203b-2025-winter`.
2. Open the project by clicking `rep-res-3rd-edition.Rproj` and compile the book by clicking `Build Book` in the `Build` panel of RStudio. (Hint: I was able to build `git_book` and `epub_book` directly. For `pdf_book`, I needed to add a line `\usepackage{hyperref}` to the file `Rep-Res-Book/rep-res-3rd-edition/latex/preabmle.tex`.)

The point of this exercise is (1) to obtain the book for free and (2) to see an example how a complicated project such as a book can be organized in a reproducible way. Use `sudo apt install PKGNAME` to install required Ubuntu packages and `tlmgr install PKGNAME` to install missing TexLive packages.

For grading purpose, include a screenshot of Section 4.1.5 of the book here.

Solution: Here is a screenshot of the book.