Biostat 203B Homework 1

Due Jan 24, 2024 @ 11:59PM

Ningke Zhang 705834790

Display machine information for reproducibility:

```
sessionInfo()
```

```
R version 4.4.2 (2024-10-31)
Platform: aarch64-apple-darwin20
Running under: macOS Sequoia 15.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.4-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_4.4.2    fastmap_1.2.0     cli_3.6.3         tools_4.4.2
 [5] htmltools_0.5.8.1 rstudioapi_0.17.1 yaml_2.3.10       rmarkdown_2.29
 [9] knitr_1.49        jsonlite_1.8.9    xfun_0.50         digest_0.6.37
[13] rlang_1.1.4       evaluate_1.0.1
```

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).

Solution: Done.

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.

Solution: Done.

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.

Solution: Develop branch created.

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.

Solution: ok.

5. After this course, you can make this repository public and use it to demonstrate your skill sets on job market.

Solution: cool.

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 link to my Completion report, and link to my Completion Certificate.

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 56
-rw-rw-r--@  1 ningkezhang  staff  15199 Oct 10 13:29 CHANGELOG.txt
-rw-rw-r--@  1 ningkezhang  staff   2518 Oct 10 14:30 LICENSE.txt
-rw-rw-r--@  1 ningkezhang  staff   2884 Oct 11 14:55 SHA256SUMS.txt
drwxr-xr-x@ 25 ningkezhang  staff    800 Jan 15 16:31 hosp
drwxr-xr-x@ 12 ningkezhang  staff    384 Jan 15 16:31 icu
-rw-rw-r--@  1 ningkezhang  staff    789 Dec 28 18:04 index.html
```

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.

Solution: I downloaded MIMIC v3.1 to my pc and made it available at `~/mimic/`.

Use Bash commands to answer following questions.

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: MIMIC-IV containing tons of real hospital stays for patients admitted, `.csv.gz` is more friendly dealing with big data, which can reduce file size, etc.

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

```
total 12306256
-rw-rw-r--@ 1 ningkezhang  staff    19928140 Jun 24  2024 admissions.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff      427554 Apr 12  2024 d_hcpcs.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff      876360 Apr 12  2024 d_icd_diagnoses.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff      589186 Apr 12  2024 d_icd_procedures.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff       13169 Oct  3 06:07 d_labitems.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff    33564802 Oct  3 06:07 diagnoses_icd.csv.gz
```

```
-rw-rw-r--@ 1 ningkezhang  staff     9743908 Oct  3 06:07 drgcodes.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff   811305629 Apr 12  2024 emar.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff   748158322 Apr 12  2024 emar_detail.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff     2162335 Apr 12  2024 hcpcsevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff        2907 Dec 28 18:04 index.html
-rw-rw-r--@ 1 ningkezhang  staff  2592909134 Oct  3 06:08 labevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff   117644075 Oct  3 06:08 microbiologyevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff    44069351 Oct  3 06:08 omr.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff     2835586 Apr 12  2024 patients.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff   525708076 Apr 12  2024 pharmacy.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff   666594177 Apr 12  2024 poe.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff    55267894 Apr 12  2024 poe_detail.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff   606298611 Apr 12  2024 prescriptions.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff     7777324 Apr 12  2024 procedures_icd.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff      127330 Apr 12  2024 provider.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff     8569241 Apr 12  2024 services.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff    46185771 Oct  3 06:08 transfers.csv.gz
total 8506792
-rw-rw-r--@ 1 ningkezhang  staff       41566 Apr 12  2024 caregiver.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff  3502392765 Apr 12  2024 chartevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff       58741 Apr 12  2024 d_items.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff    63481196 Apr 12  2024 datetimeevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff     3342355 Oct  3 04:36 icustays.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff        1336 Dec 28 18:04 index.html
-rw-rw-r--@ 1 ningkezhang  staff   311642048 Apr 12  2024 ingredientevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff   401088206 Apr 12  2024 inputevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff    49307639 Apr 12  2024 outputevents.csv.gz
-rw-rw-r--@ 1 ningkezhang  staff    24096834 Apr 12  2024 procedureevents.csv.gz
```

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

Solution:
`zcat` is used to display the content of `.csv.gz` compressed files without uncompressing files permanently. `zless` is used to view the content of a compressed file in scrolling way. `zmore`also displaying the compressed file content page by page. `zgrep` can search a specific word in the compressed file.

```
ls ~/mimic/hosp/admissions.csv.gz
zcat < ~/mimic/hosp/admissions.csv.gz | head -5
zless ~/mimic/hosp/admissions.csv.gz | head -5
zmore ~/mimic/hosp/admissions.csv.gz | head -5
zgrep "admission_type" ~/mimic/hosp/admissions.csv.gz
```

```
/Users/ningkezhang/mimic/hosp/admissions.csv.gz
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
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
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
subject_id,hadm_id,admittime,dischtime,deathtime,admission_type,admit_provider_id,admission_l
```

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

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

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

Solution: The output is file permissions, number of links, owner name, owner group, file size, date and time of last modification, and file name/path.

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

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: The first few lines of `admissions.csv.gz` are displayed below. There are 546028 rows in this data file, excluding the header line. There are 546029 hospitalizations in this data file. There are 223453 unique patients in this data file. The number of patients listed in the `patients.csv.gz` file is 364627. Does not match.

```
zcat < ~/mimic/hosp/admissions.csv.gz | head -5

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

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

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

```
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
  546028
  546028
  223452
  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: The possible values taken by each of the variable and count for each unique value in decreasing order are displayed below.

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

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

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

```
zcat < ~/mimic/hosp/admissions.csv.gz | tail -n +2 |
  awk -F, '{print $13}' | 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
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
244576 Medicare
173399 Private
104229 Medicaid
14006 Other
9355
 463 No charge
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: There are 994458 ICU stays in this data file. There are 65366 unique patients in this data file.

```
zcat < ~/mimic/icu/icustays.csv.gz | head -1

zcat < ~/mimic/icu/icustays.csv.gz | tail -n +2 |
  awk -F, '{print $3}' | wc -l
zcat < ~/mimic/icu/icustays.csv.gz | tail -n +2 |
  awk -F, '{print $1}' | sort | uniq | wc -l
```

```
subject_id,hadm_id,stay_id,first_careunit,last_careunit,intime,outtime,los
   94458
   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: Comressed file size is 2.4G, uncompressed file size is 17G. The run time of `zcat < ~/mimic/labevents.csv.gz | wc -l` is 15.454s, and the run time of `wc -l labevents.csv` is 15.537s. The trade off between storage and speed for big data files is that compressed files save storage space but take longer to process, while uncompressed files take up more storage space but are faster to process. This makes compressed files more suitable for long-term storage. (setting no output for this chunk)

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
```

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` downloads the file, -nc to avoid downloading the file if it already exists. Elizabeth: 634, Jane: 293, Lydia: 171, Darcy: 418.

```
wget -nc http://www.gutenberg.org/cache/epub/42671/pg42671.txt
for char in Elizabeth Jane Lydia Darcy
do
  echo $char:
  grep -o -i $char /Users/ningkezhang/Downloads/pg42671.txt | wc -l
done
```

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

Solution: The first command, '>' writes the output to a new file `test1.txt`, while the second command, '»' appends the output to an existing file `test2.txt`.

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

and

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

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
```

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: `"$1"` here is the first argument, which is the file pg42671.txt. `"$2"` is the second argument, which is 20, and `"$3"` is the third argument, for here is 5. `head -n "$2" "$1` is selecting the first 20 lines of the file `pg42671.txt`, and `tail -n "$3"` is selecting the last 5 lines of the first 20 lines. The first line of the shell script is a shebang, that tells the system which interpreter to use to run the script.

```
echo '#!/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"' > middle.sh

chmod u+x middle.sh
./middle.sh pg42671.txt 20 5
```

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

Language: English

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` displays the calendar for the current month. `cal 2025` displays the calendar for the year 2025. `cal 9 1752` displays the calendar for September 1752, but the calendar is incorrect because the Gregorian calendar was adopted in 1752. `date` displays the current date and time. `hostname` displays the name of the host. `arch` displays the architecture of the system. `uname -a` displays the system information. `uptime` displays the system uptime. `who am i` displays the current user. `who` displays the users currently logged in. `w` displays the users currently logged in and what they are doing. `id` displays the user and group information. `last | head` displays the last login information. `echo {con,pre}{sent,fer}{s,ed}` displays the combinations of the words. `time sleep 5` displays the time it takes to sleep for 5 seconds. `history | tail` displays the last few commands in the history.

```
cal
cal 2025
cal 9 1752
date
uname -a
uptime
who am i
who
w
id
last | head
echo {con,pre}{sent,fer}{s,ed}
time sleep 5
history | tail
```

```
    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
```

```
                              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

   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



Thu Jan 23 18:08:23 PST 2025
Darwin Mac.attlocal.net 24.1.0 Darwin Kernel Version 24.1.0: Thu Oct 10 21:02:26 PDT 2024; r
```

```
18:08  up 7 days, 17:08, 1 user, load averages: 1.84 2.01 1.90
ningkezhang                    Jan 23 18:08
ningkezhang      console       Jan 21 00:41
18:08  up 7 days, 17:08, 1 user, load averages: 1.84 2.01 1.90
USER       TTY      FROM    LOGIN@  IDLE WHAT
ningkezhan console  -      Tue00   2days -
uid=501(ningkezhang) gid=20(staff) groups=20(staff),12(everyone),61(localaccounts),79(_appse
ningkezhang ttys001                          Tue Jan 21 00:52 - 00:52  (00:00)
ningkezhang ttys001                          Tue Jan 21 00:42 - 00:42  (00:00)
ningkezhang console                          Tue Jan 21 00:41   still logged in
ningkezhang ttys000                          Mon Jan 20 23:39 - 23:39  (00:00)
ningkezhang ttys000                          Mon Jan 20 22:25 - 22:25  (00:00)
ningkezhang ttys001                          Mon Jan 20 18:46 - 18:46  (00:00)
ningkezhang ttys001                          Sun Jan 19 01:37 - 01:37  (00:00)
ningkezhang ttys000                          Thu Jan 16 19:52 - 19:52  (00:00)
ningkezhang ttys000                          Thu Jan 16 18:36 - 18:36  (00:00)
ningkezhang console                          Thu Jan 16 01:01 - 00:41 (4+23:40)
consents consented confers confered presents presented prefers prefered

real    0m5.008s
user    0m0.001s
sys 0m0.001s
```

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 the screenshot of Section 4.1.5.