Biostat 203B Homework 1

Due Jan 24, 2025 @ 11:59PM

Wenjing Zhou and 806542441

Display machine information for reproducibility:

```
sessionInfo()
```

```
R version 4.4.1 (2024-06-14)
Platform: x86_64-apple-darwin20
Running under: macOS 15.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.4-x86_64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.4-x86_64/Resources/lib/libRlapack.dylib;

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_4.4.1    fastmap_1.2.0     cli_3.6.3         tools_4.4.1
 [5] htmltools_0.5.8.1 rstudioapi_0.17.0 yaml_2.3.10       rmarkdown_2.29
 [9] knitr_1.48        jsonlite_1.8.9    xfun_0.48         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,

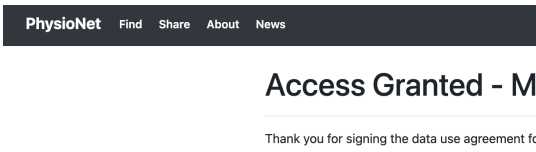
is an important criterion for grading your homework.

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).
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.I've already created a repository named `biostat-203b-2025-winter` and here is the repository link.

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 Done. Here is the screenshot of mimic data access approval.

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 vickey  staff  15199 Oct 10 16:29 CHANGELOG.txt
-rw-r--r--@  1 vickey  staff   2518 Oct 10 17:30 LICENSE.txt
-rw-r--r--@  1 vickey  staff   2884 Oct 11 17:55 SHA256SUMS.txt
drwxr-xr-x@ 23 vickey  staff    736 Jan 24 03:06 hosp
drwxr-xr-x@ 11 vickey  staff    352 Jan 22 16:08 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.

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: Display the contents

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

```
total 10845000
-rw-r--r--@ 1 vickey  staff    19928140 Jun 24  2024 admissions.csv.gz
-rw-r--r--@ 1 vickey  staff      427554 Apr 12  2024 d_hcpcs.csv.gz
-rw-r--r--@ 1 vickey  staff      876360 Apr 12  2024 d_icd_diagnoses.csv.gz
-rw-r--r--@ 1 vickey  staff      589186 Apr 12  2024 d_icd_procedures.csv.gz
-rw-r--r--@ 1 vickey  staff       13169 Oct  3 09:07 d_labitems.csv.gz
-rw-r--r--@ 1 vickey  staff    33564802 Oct  3 09:07 diagnoses_icd.csv.gz
-rw-r--r--@ 1 vickey  staff     9743908 Oct  3 09:07 drgcodes.csv.gz
-rw-r--r--@ 1 vickey  staff   811305629 Apr 12  2024 emar.csv.gz
-rw-r--r--@ 1 vickey  staff     2162335 Apr 12  2024 hcpcsevents.csv.gz
```

```
-rw-r--r--@ 1 vickey  staff  2592909134 Oct  3 09:08 labevents.csv.gz
-rw-r--r--@ 1 vickey  staff   117644075 Oct  3 09:08 microbiologyevents.csv.gz
-rw-r--r--@ 1 vickey  staff    44069351 Oct  3 09:08 omr.csv.gz
-rw-r--r--@ 1 vickey  staff     2835586 Apr 12  2024 patients.csv.gz
-rw-r--r--@ 1 vickey  staff   525708076 Apr 12  2024 pharmacy.csv.gz
-rw-r--r--@ 1 vickey  staff   666594177 Apr 12  2024 poe.csv.gz
-rw-r--r--@ 1 vickey  staff    55267894 Apr 12  2024 poe_detail.csv.gz
-rw-r--r--@ 1 vickey  staff   606298611 Apr 12  2024 prescriptions.csv.gz
-rw-r--r--@ 1 vickey  staff     7777324 Apr 12  2024 procedures_icd.csv.gz
-rw-r--r--@ 1 vickey  staff      127330 Apr 12  2024 provider.csv.gz
-rw-r--r--@ 1 vickey  staff     8569241 Apr 12  2024 services.csv.gz
-rw-r--r--@ 1 vickey  staff    46185771 Oct  3 09:08 transfers.csv.gz
total 8506784
-rw-r--r--@ 1 vickey  staff       41566 Apr 12  2024 caregiver.csv.gz
-rw-r--r--@ 1 vickey  staff  3502392765 Apr 12  2024 chartevents.csv.gz
-rw-r--r--@ 1 vickey  staff       58741 Apr 12  2024 d_items.csv.gz
-rw-r--r--@ 1 vickey  staff    63481196 Apr 12  2024 datetimeevents.csv.gz
-rw-r--r--@ 1 vickey  staff     3342355 Oct  3 07:36 icustays.csv.gz
-rw-r--r--@ 1 vickey  staff   311642048 Apr 12  2024 ingredientevents.csv.gz
-rw-r--r--@ 1 vickey  staff   401088206 Apr 12  2024 inputevents.csv.gz
-rw-r--r--@ 1 vickey  staff    49307639 Apr 12  2024 outputevents.csv.gz
-rw-r--r--@ 1 vickey  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 contents of a compressed file. It is equivalent to `cat` command for uncompressed files.

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

```
subject_id,hadm_id,stay_id,first_careunit,last_careunit,intime,outtime,los
10000032,29079034,39553978,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10000690,25860671,37081114,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10000980,26913865,39765666,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10001217,24597018,37067082,Surgical Intensive Care Unit (SICU),Surgical Intensive Care Unit
10001217,27703517,34592300,Surgical Intensive Care Unit (SICU),Surgical Intensive Care Unit
10001725,25563031,31205490,Medical/Surgical Intensive Care Unit (MICU/SICU),Medical/Surgical
10001843,26133978,39698942,Medical/Surgical Intensive Care Unit (MICU/SICU),Medical/Surgical
10001884,26184834,37510196,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10002013,23581541,39060235,Cardiac Vascular Intensive Care Unit (CVICU),Cardiac Vascular Inte
```

- `zless` is used to display the contents of a compressed file one screen at a time. It is equivalent to `less` command for uncompressed files.

```
zless ~/mimic/hosp/d_hcpcs.csv.gz | head
```

```
code,category,long_description,short_description
00000,,,Invalid Code
0001F,2,,Composite measures
0002F,2,,Composite measures
0003F,2,,Composite measures
0004F,2,,Composite measures
0005F,2,,Composite measures
0006F,2,,Composite measures
0007F,2,,Composite measures
0008F,2,,Composite measures
```

- `zmore` is used to display the contents of a compressed file one screen at a time. It is equivalent to `more` command for uncompressed files.

```
zmore ~/mimic/hosp/d_hcpcs.csv.gz | head
```

```
code,category,long_description,short_description
00000,,,Invalid Code
0001F,2,,Composite measures
0002F,2,,Composite measures
0003F,2,,Composite measures
0004F,2,,Composite measures
0005F,2,,Composite measures
0006F,2,,Composite measures
0007F,2,,Composite measures
0008F,2,,Composite measures
```

- `zgrep` is used to search the contents of a compressed file using regular expressions. It is equivalent to `grep` command for uncompressed files.

```
zgrep -c "99291" ~/mimic/hosp/services.csv.gz
```

```
32
```

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

Solution

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

The output of this bash is displaying all the file names statring with `a`, `l`, and `pa` in the `hosp` folder.

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

Solution:

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

```
  546029
 158374765
  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.)

```
# display the first few lines
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 HOSPI
10000032,22841357,2180-06-26 18:27:00,2180-06-27 18:49:00,,EW EMER.,P784FA,EMERGENCY ROOM,HO
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,HO
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 REFE
10000084,29888819,2160-12-28 05:11:00,2160-12-28 16:07:00,,EU OBSERVATION,P35NE4,PHYSICIAN R
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
```

```
# count the number of rows (excluding the header line)
zcat < ~/mimic/hosp/admissions.csv.gz |tail -n +2 | wc -l
```

```
  546028
```

```
# count the number of unique hospitalizations
zcat < ~/mimic/hosp/admissions.csv.gz | tail -n +2 | awk -F, '{print $2}' | sort | uniq | wc
```

```
  546028
```

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

```
  223452
```

```
# display the first few lines of patients.csv.gz
zcat < ~/mimic/hosp/patients.csv.gz | head
# count the number of unique patients in patients.csv.gz
zcat < ~/mimic/hosp/patients.csv.gz | tail -n +2 | awk -F, '{print $1}' | sort | uniq | wc -
```

```
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,
  364627
```

Therefore the number of unique patients in `admissions.csv.gz` and `patients.csv.gz` are not the same.

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:

```
# display 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 HOSPI
10000032,22841357,2180-06-26 18:27:00,2180-06-27 18:49:00,,EW EMER.,P784FA,EMERGENCY ROOM,HO
10000032,25742920,2180-08-05 23:44:00,2180-08-07 17:50:00,,EW EMER.,P19UTS,EMERGENCY ROOM,HO
10000032,29079034,2180-07-23 12:35:00,2180-07-25 17:55:00,,EW EMER.,P06OTX,EMERGENCY ROOM,HO
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 REFE
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
```

```
# count the number of unique values for admission_type
zcat < ~/mimic/hosp/admissions.csv.gz | tail -n +2 | awk -F, '{print $6}' | sort | uniq -c |
```

```
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
```

```
# count the number of unique values for admission_location
zcat < ~/mimic/hosp/admissions.csv.gz | tail -n +2 | awk -F, '{print $8}' | sort | uniq -c |
```

```
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
```

```
# count the number of unique values for insurance
zcat < ~/mimic/hosp/admissions.csv.gz | tail -n +2 | awk -F, '{print $10}' | sort | uniq -c
```

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

```
# count the number of unique values
zcat < ~/mimic/hosp/admissions.csv.gz | tail -n +2 | awk -F, '{print $13}' | sort | uniq -c
```

```
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 `icusays.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:

```
# display the first few lines
zcat < ~/mimic/icu/icustays.csv.gz | head
# the number of ICU stays
zcat < ~/mimic/icu/icustays.csv.gz | tail -n +2 | awk -F, '{print $3}' | sort | uniq | wc -l
# the number of unique patients
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
10000032,29079034,39553978,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10000690,25860671,37081114,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10000980,26913865,39765666,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10001217,24597018,37067082,Surgical Intensive Care Unit (SICU),Surgical Intensive Care Unit
10001217,27703517,34592300,Surgical Intensive Care Unit (SICU),Surgical Intensive Care Unit
10001725,25563031,31205490,Medical/Surgical Intensive Care Unit (MICU/SICU),Medical/Surgical
10001843,26133978,39698942,Medical/Surgical Intensive Care Unit (MICU/SICU),Medical/Surgical
10001884,26184834,37510196,Medical Intensive Care Unit (MICU),Medical Intensive Care Unit (MI
10002013,23581541,39060235,Cardiac Vascular Intensive Care Unit (CVICU),Cardiac Vascular Inte
   94458
   65366
```

The number of ICU stays is 94458 and the number of unique patients is 65366 in this data file.

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:

```
# compare compressed and uncompressed file size
ls -lh ~/mimic/hosp/labevents.csv.gz
# Uncompress the file temporarily and check its size
gzip -dk < ~/mimic/hosp/labevents.csv.gz > ./labevents.csv
ls -lh labevents.csv
```

```
-rw-r--r--@ 1 vickey  staff   2.4G Oct  3 09:08 /Users/vickey/mimic/hosp/labevents.csv.gz
-rw-r--r--@ 1 vickey  staff    17G Jan 24 14:23 labevents.csv
```

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

```
 158374765

real    0m35.699s
user    0m51.265s
sys 0m3.174s
 158374765 ./labevents.csv

real    0m39.578s
user    0m34.726s
sys 0m4.338s
```

The compressed file size is smaller than the uncompressed file size. The run time of `zcat < ~/mimic/labevents.csv.gz | wc -l` is faster than `wc -l ~/mimic/labevents.csv`. The trade off between storage and speed for big data files is that compressed files save storage space but take longer time to read and write, while uncompressed files take more storage space but are faster to read and write.

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` is used to download the file only if it does not exist in the current directory.

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

for char in Elizabeth Jane Lydia Darcy
do
  echo $char:
  # some bash commands here
   grep -oi "$char" pg42671.txt | wc -l
done
```

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

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

and

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

Solution: The difference of the two commands is that the first command `echo 'hello, world' > test1.txt` writes the output to a file `test1.txt` and overwrites the file if it already exists, while the second command `echo 'hello, world' >> test2.txt` appends the output to a file `test2.txt` if the file already exists. `test2.txt` will have the content `hello, world` twice if I run the second command twice.

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

```
chmod +x middle.sh
```

```
./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: The meaning of `"$1"`, `"$2"`, and `"$3"` in this shell script is the first, second, and third argument passed to the script, respectively. The first line of the shell script `#!/bin/sh` is called a shebang line, which tells the system which interpreter to use to execute the script. In this case, it tells the system to use the `sh` interpreter to execute the script.

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
```

Unusual thing: The days from 3rd to 13th on Sepetember of 1752 in the calnedar are missing.

```
date
```

```
Fri Jan 24 14:24:52 PST 2025
```

```
hostname
```

```
MacBookPro.lan
```

```
arch
```

```
i386
```

```
uname -a
```

```
Darwin MacBookPro.lan 24.1.0 Darwin Kernel Version 24.1.0: Thu Oct 10 21:02:27 PDT 2024; roo
```

```
uptime
```

```
14:24  up 11:21, 1 user, load averages: 2.98 2.64 2.50
```

```
who am i
```

```
vickey                       Jan 24 14:24
```

```
who
```

```
vickey           console      Jan 24 03:04
```

```
w
```

```
14:24  up 11:21, 1 user, load averages: 2.98 2.64 2.50
USER       TTY      FROM    LOGIN@  IDLE WHAT
vickey     console  -       3:04   11:20 -
```

```
id
```

```
uid=501(vickey) gid=20(staff) groups=20(staff),101(access_bpf),12(everyone),61(localaccounts)
```

```
last | head
```

```
wtmp begins Fri Jan 24 14:24:52 PST 2025
```

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

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

```
time sleep 5
```

```
real    0m5.011s
user    0m0.002s
sys 0m0.006s
```

```
history | tail
```

9 brew install wget 10 history | tail 11 git branch 12 git check out main 13 git checkout main 14 ls 15 git branch 16 git checkout main 17 git merge develop 18 history | tail

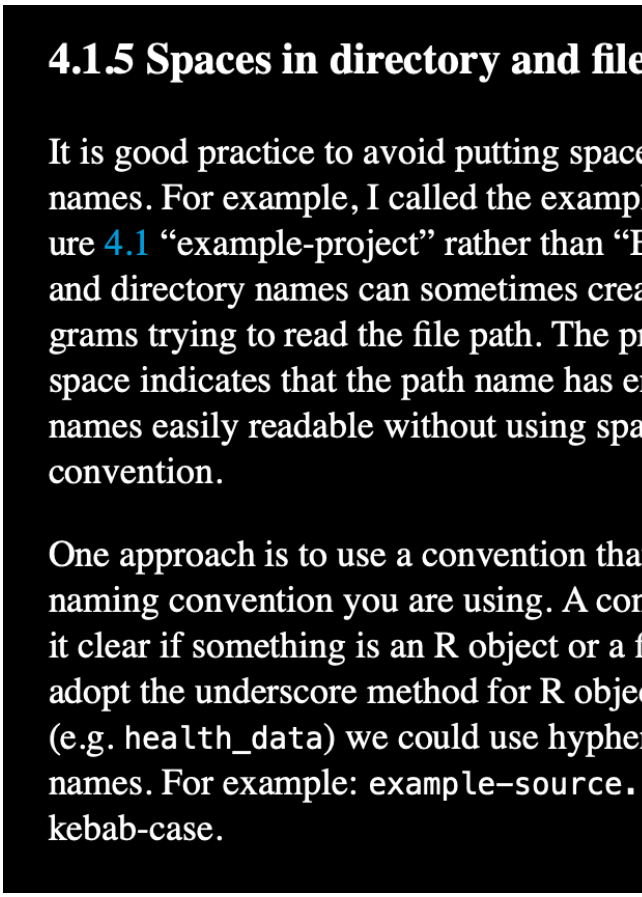
Q6. Book

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 of the book.



4.1.5 Spaces in directory and file

It is good practice to avoid putting space
names. For example, I called the examp
ure 4.1 "example-project" rather than "E
and directory names can sometimes crea
grams trying to read the file path. The pr
space indicates that the path name has e
names easily readable without using spa
convention.

One approach is to use a convention tha
naming convention you are using. A con
it clear if something is an R object or a f
adopt the underscore method for R objec
(e.g. `health_data`) we could use hyphe
names. For example: `example-source.`
kebab-case.