Biostat 203B Homework 1

Due Jan 24, 2025 @ 11:59PM

Luke Hodges 906182810

Display machine information for reproducibility:

```
sessionInfo()
```

```
R version 4.4.2 (2024-10-31)
Platform: x86_64-apple-darwin20
Running under: macOS Sequoia 15.0

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_4.4.2    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.

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).

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.

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.

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.

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 my Completion Certificate of my CITI Training.

Q3. Linux Shell Commands

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

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

```
total 48
-rw-r--r--@  1 lukehodges  staff  15199 Oct 10 16:29 CHANGELOG.txt
-rw-r--r--@  1 lukehodges  staff   2518 Oct 10 17:30 LICENSE.txt
-rw-r--r--@  1 lukehodges  staff   2884 Oct 11 17:55 SHA256SUMS.txt
drwxr-xr-x@ 26 lukehodges  staff    832 Jan 20 22:00 hosp
drwxr-xr-x@ 11 lukehodges  staff    352 Jan 20 18:41 icu
lrwxr-xr-x   1 lukehodges  staff     38 Jan 20 19:35 mimic-iv-3.1 -> /Users/lukehodges/Deskto
```

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 is available under the `~/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 the `hosp` folder

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

```
total 48249320
-rw-r--r--@ 1 lukehodges  staff     19928140 Jun 24  2024 admissions.csv.gz
-rw-r--r--@ 1 lukehodges  staff       427554 Apr 12  2024 d_hcpcs.csv.gz
-rw-r--r--@ 1 lukehodges  staff       876360 Apr 12  2024 d_icd_diagnoses.csv.gz
-rw-r--r--@ 1 lukehodges  staff       589186 Apr 12  2024 d_icd_procedures.csv.gz
-rw-r--r--@ 1 lukehodges  staff        13169 Oct  3 09:07 d_labitems.csv.gz
-rw-r--r--@ 1 lukehodges  staff     33564802 Oct  3 09:07 diagnoses_icd.csv.gz
-rw-r--r--@ 1 lukehodges  staff      9743908 Oct  3 09:07 drgcodes.csv.gz
-rw-r--r--@ 1 lukehodges  staff    811305629 Apr 12  2024 emar.csv.gz
-rw-r--r--@ 1 lukehodges  staff    748158322 Apr 12  2024 emar_detail.csv.gz
-rw-r--r--@ 1 lukehodges  staff      2162335 Apr 12  2024 hcpcsevents.csv.gz
-rw-r--r--@ 1 lukehodges  staff  18402851720 Jan 20 22:00 labevents.csv
-rw-r--r--@ 1 lukehodges  staff   2592909134 Oct  3 09:08 labevents.csv.gz
```

```
-rw-r--r--@ 1 lukehodges  staff    117644075 Oct  3 09:08 microbiologyevents.csv.gz
-rw-r--r--@ 1 lukehodges  staff     44069351 Oct  3 09:08 omr.csv.gz
-rw-r--r--@ 1 lukehodges  staff      2835586 Apr 12  2024 patients.csv.gz
-rw-r--r--@ 1 lukehodges  staff    525708076 Apr 12  2024 pharmacy.csv.gz
-rw-r--r--@ 1 lukehodges  staff    666594177 Apr 12  2024 poe.csv.gz
-rw-r--r--@ 1 lukehodges  staff     55267894 Apr 12  2024 poe_detail.csv.gz
-rw-r--r--@ 1 lukehodges  staff    606298611 Apr 12  2024 prescriptions.csv.gz
-rw-r--r--@ 1 lukehodges  staff      7777324 Apr 12  2024 procedures_icd.csv.gz
-rw-r--r--@ 1 lukehodges  staff       127330 Apr 12  2024 provider.csv.gz
-rw-r--r--@ 1 lukehodges  staff      8569241 Apr 12  2024 services.csv.gz
-rw-r--r--@ 1 lukehodges  staff     46185771 Oct  3 09:08 transfers.csv.gz
```

Solution Here is the content of the `icu` folder

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

```
total 8506784
-rw-r--r--@ 1 lukehodges  staff       41566 Apr 12  2024 caregiver.csv.gz
-rw-r--r--@ 1 lukehodges  staff  3502392765 Apr 12  2024 chartevents.csv.gz
-rw-r--r--@ 1 lukehodges  staff       58741 Apr 12  2024 d_items.csv.gz
-rw-r--r--@ 1 lukehodges  staff    63481196 Apr 12  2024 datetimeevents.csv.gz
-rw-r--r--@ 1 lukehodges  staff     3342355 Oct  3 07:36 icustays.csv.gz
-rw-r--r--@ 1 lukehodges  staff   311642048 Apr 12  2024 ingredientevents.csv.gz
-rw-r--r--@ 1 lukehodges  staff   401088206 Apr 12  2024 inputevents.csv.gz
-rw-r--r--@ 1 lukehodges  staff    49307639 Apr 12  2024 outputevents.csv.gz
-rw-r--r--@ 1 lukehodges  staff    24096834 Apr 12  2024 procedureevents.csv.gz
```

These data files were distrbuted as gz files because they are giant files and allows them to be downloaded and distributed faster and easier.

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

Solution Zcat decompresses a compressed file (.gz).

Solution Zmore views compressed files, but cannot navigate as easily

Solution Zless views compressed files with easy navigation line by line

Solution Zgrep searches compressed files for terms and returns with matching lines.

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

Solution Here is the output

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

```
-rw-r--r--@ 1 lukehodges  staff  19928140 Jun 24  2024 /Users/lukehodges/mimic/hosp/admissio
-rw-r--r--@ 1 lukehodges  staff  2592909134 Oct  3 09:08 /Users/lukehodges/mimic/hosp/labeve
-rw-r--r--@ 1 lukehodges  staff  2835586 Apr 12  2024 /Users/lukehodges/mimic/hosp/patients.
```

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

Solution Here are the lines in admissions file

```
for datafile in ~/mimic/hosp/admissions.csv.gz
do
zcat < $datafile | wc -l
done
```

```
  546029
```

Solution Here are the lines in the labevents file

```
for datafile in ~/mimic/hosp/labevents.csv.gz
do
zcat < $datafile | wc -l
done
```

```
 158374765
```

Solution Here are the lines in the patients file

```
for datafile in ~/mimic/hosp/patients.csv.gz
do
zcat < $datafile | wc -l
done
```

```
  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 are the first few lines of admissions.csv.gz

```
for datafile in ~/mimic/hosp/admissions.csv.gz
do
zcat < $datafile | head
done
```

```
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.,P060TX,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
```

Solution Counting the total number of rows excluding the header

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

```
  546028
```

Solution Number of unique hospitalizations

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

```
  546028
```

The number of rows is the number of unique hospitalizations.

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 file is:

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

```
  223452
```

This should match the number in the patients file

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

```
  364627
```

The total number of unique patients in the admissions file is less.

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 You need to first examine the admissions file.

```
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
```

Admission type is the sixth column so the possible values include:

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

```
       9
```

So there are nine different admission types.

For Admissions location:

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

```
      12
```

There are twelve different admission locations.

For insurance:

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

```
       6
```

There are six insurance

For ethnicity:

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

```
      33
```

There are 33 ethnicities.

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

To start, we must look into the icusays.csv.gz file

```
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
```

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
```

There are 94458 ICU stays identified by the stay_id

The number of patients identified by subject_id

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

```
  223452
```

There are 223452 patients identified by subject_id

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.)

Checking file size for the compressed version

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

```
-rw-r--r--@ 1 lukehodges  staff   2.4G Oct  3 09:08 /Users/lukehodges/mimic/hosp/labevents.cs
```

This file is 2.4G

Checking file size for the uncompressed version

```
ls -lh ~/mimic/hosp/labevents.csv
```

```
-rw-r--r--@ 1 lukehodges  staff    17G Jan 20 22:00 /Users/lukehodges/mimic/hosp/labevents.cs
```

This file is 17G

Comparing the run times of the compressed and uncompressed

For Compressed

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

```
 158374765

real    0m52.504s
user    1m15.768s
sys 0m4.668s
```

For Uncompressed

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

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

real    0m57.912s
user    0m50.990s
sys 0m5.536s
```

This shows that compressed is slower than uncompressed.

This is because compressed has to go through the decompressing of the file

In other words, compressed saves storage, but slows time

Uncompressed takes up storage, but is faster.

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
for char in Elizabeth Jane Lydia Darcy
do
  echo $char:
  grep -n "$char" pg42671.txt | wc -l
done
```

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

Elizabeth:
     633
Jane:
     289
Lydia:
     166
Darcy:
     414
```

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.

Wget -nc downloads the file, but doers not replicate if it already is downloaded.

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

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

and

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

The first command directs the output to the test1.txt, saying ‘hello, world’.

The second command appends (adds it) to a file named test2.txt.

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

The middle.sh is set to my desktop. The command chmod 751 gives executive function and access to the user.

```
chmod 751 ~/middle.sh
```

```
~/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?

The output is the 20th line of the file pg42671.txt and the following 5 lines. The “$1” is the first argument, which is the file name. The “$2” is the second argument, which is the line. The “$3” is the third argument, which is the number of lines. The first line of the shell script is needed to tell the computer what shell to use to run 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.

```
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
```

This makes a calendar of the current month of the current year

```
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
```

This makes an annual calendar

```
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
```

This makes a calendar for September, 1752. It skips 3-13.

```
date
```

```
Fri Jan 24 12:07:05 PST 2025
```

Gives you current date and time

```
hostname
```

```
dhcp-10-144-183-115.dgsom.guest
```

Gives name of local computer

```
arch
```

```
i386
```

Indicates the architecture of the computer (32-bit system)

```
uname -a
```

```
Darwin dhcp-10-144-183-115.dgsom.guest 24.0.0 Darwin Kernel Version 24.0.0: Mon Aug 12 20:54
```

Gives you more details of the local computer system, like version.

```
uptime
```

```
12:07  up 4 days, 19:18, 1 user, load averages: 2.97 2.87 3.11
```

Shows you how long the computer has been running

```
who am i
```

```
lukehodges                    Jan 24 12:07
```

Gives you name of the user with date and time

```
who
```

```
lukehodges       console      Jan 19 16:49
```

Gives you who is currently using the local computer.

```
w
```

```
12:07  up 4 days, 19:18, 1 user, load averages: 2.97 2.87 3.11
USER       TTY      FROM    LOGIN@  IDLE WHAT
lukehodges console  -      Sun16   4days -
```

Seems to summarize and add from the information above.

```
id
```

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

Displays individual and group IDs

```
last | head
```

```
lukehodges ttys000                        Fri Jan 24 11:12 - 11:12  (00:00)
lukehodges ttys000                        Thu Jan 23 17:58 - 17:58  (00:00)
lukehodges ttys000                        Tue Jan 21 12:26 - 12:26  (00:00)
lukehodges ttys000                        Tue Jan 21 12:23 - 12:23  (00:00)
lukehodges ttys001                        Mon Jan 20 22:34 - 22:34  (00:00)
lukehodges ttys001                        Mon Jan 20 19:33 - 19:33  (00:00)
lukehodges ttys001                        Mon Jan 20 18:46 - 18:46  (00:00)
lukehodges ttys001                        Mon Jan 20 18:45 - 18:45  (00:00)
lukehodges ttys000                        Mon Jan 20 18:44 - 18:44  (00:00)
lukehodges ttys001                        Mon Jan 20 18:42 - 18:42  (00:00)
```

Displays the first few lines of people who have logged in and when

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

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

Matches every possible combination in the set in order

```
time sleep 5
```

```


real    0m5.010s
user    0m0.003s
sys 0m0.005s
```

Shows how long the sleep command takes to run

```
history | tail
```

Displays the last several commands executed

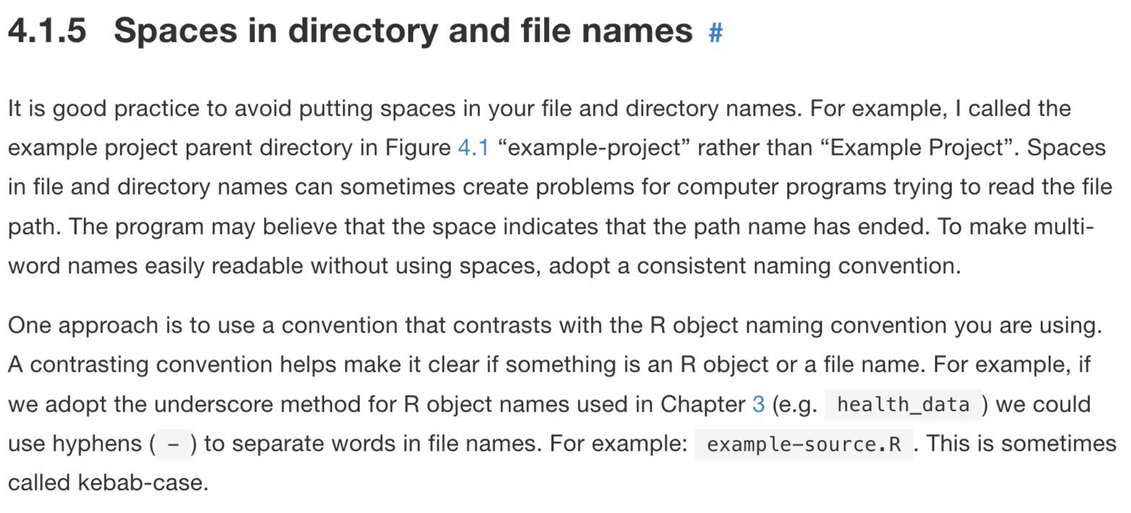
Q6. Git clone the repository https://github.com/christophergandrud/Rep-Res-Book for the book Reproducible Research with R and RStudio to your local machine. Do not put this repository within your homework repository biostat-203b-2025-winter.

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 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 picture



4.1.5 Spaces in directory and file names #

It is good practice to avoid putting spaces in your file and directory names. For example, I called the example project parent directory in Figure 4.1 "example-project" rather than "Example Project". Spaces in file and directory names can sometimes create problems for computer programs trying to read the file path. The program may believe that the space indicates that the path name has ended. To make multi-word names easily readable without using spaces, adopt a consistent naming convention.

One approach is to use a convention that contrasts with the R object naming convention you are using. A contrasting convention helps make it clear if something is an R object or a file name. For example, if we adopt the underscore method for R object names used in Chapter 3 (e.g. `health_data`) we could use hyphens (`-`) to separate words in file names. For example: `example-source.R`. This is sometimes called kebab-case.

Figure 1: Section