Knowledge Quiz 2

KEY

Please answer the following questions, render a pdf, and submit both the qmd and pdf on moodle by 11 PM on Thurs Nov 14. Please also leave a copy of your qmd in your Submit folder on the St. Olaf RStudio server.

Guidelines:

- No consulting with anyone else
- You may use only materials from this class (our class webpage, links on moodle, our 3 online textbooks, files posted to the RStudio server, your personal notes from class)
- No online searches or use of large language models like ChatGPT

Pledge:

I pledge my honor that on this quiz I have neither given nor received assistance not explicitly approved by the professor and that I an aware of no dishonest work.

- type your name here to acknowledge the pledge: _____
- OR
- place an X here if you intentionally are not signing the pledge: _____

```
library(tidyverse)
library(rvest)
library(tidytext)

park_data <- read_csv("~/264_fall_2024/DS2_preview_work/park_data_KQ2.csv")
#park_data <- read_csv("~/Sds 264 F24/Class/Data/park_data_KQ2.csv")
```

National Park Data

`park_data` is a 54x3 tibble containing information scraped from national park webpages for a past SDS264 final project. A few notes about the 3 columns:

- `park_code` is a 4-letter code used as a key when merging files
- `address` is comprised of 4 pieces (described from *right* to *left*):
  - the final piece (following a comma and space) is a zip code (usually 5 digits but sometimes 5 digits then a dash then 4 more digits)
  - the 2nd to last piece is the state (an abbreviation with 2 capital letters)
  - the 3rd to last piece is the city (usually one or two words long, occasionally 3; always follows two or more spaces)
  - the first piece is the street address (often a number and a street, but will always be followed by at least two spaces)
- `activities` is a string of activities offered at each park, where activities are separated by commas

Quiz Questions

Please answer the following questions using your knowledge of strings, regular expressions, and text analysis. Please use `stringr` functions as much as possible, aim for efficient code, and use good style to make your code as readable as possible!

Section 1

1. Find the subset of all `address` entries that contain a direction (north, south, east, or west).

```
str_subset(str_to_lower(park_data$address), "north|south|east|west")
```

```
[1] "52 west headquarters drive   torrey ut, 84775"
[2] "64 grinnell drive  west glacier mt, 59936"
[3] "20 south entrance road  grand canyon az, 86023"
[4] "800 east lakeshore drive   houghton mi, 49931"
[5] "38050 highway 36 east  mineral ca, 96063"
[6] "55210 238th avenue east   ashford wa, 98304"
[7] "5000 east entrance road   paicines ca, 95043"
[8] "3655 u.s. highway 211  east luray va, 22835"
[9] "360 hwy 11 east  international falls mn, 56649"
```

2. Produce a tibble showing how often each of the 4 directions from (1) occurs among the 54 `address` entries. Which direction is most common?

```
park_data |>
  mutate(
    address_low = str_to_lower(address),
    north = str_count(address_low, "north"),
    south = str_count(address_low, "south"),
    east = str_count(address_low, "east"),
    west = str_count(address_low, "west")
  ) |>
  summarize(
    north = sum(north),
    south = sum(south),
    east = sum(east),
    west = sum(west)
  )
```

```
# A tibble: 1 x 4
  north south  east  west
  <int> <int> <int> <int>
1     0     1     6     2
```

3. Create a new tibble containing only national parks in Alaska (AK) and Hawaii (HI).

```
park_data |>
  filter(str_detect(address, "AK|HI"))
```

```
# A tibble: 10 x 3
   park_code address                                                activities
   <chr>     <chr>                                                  <chr>
 1 DENA      Mile 237 Highway 3   Denali Park AK, 99755             Arts and Cu~
 2 GAAR      101 Dunkel St   Fairbanks AK, 99701                    Camping, Ba~
 3 GLBA      1 Park Road   Gustavus AK, 99826                       Arts and Cu~
 4 HALE      Haleakala National Park Route 378  Kula HI, 96790      Camping, Ba~
 5 HAVO      1 Crater Rim Drive   Hawaii National Park HI, 96718    Arts and Cu~
 6 KATM      1000 Silver Street  King Salmon AK, 99613              Boating, Ca~
 7 KEFJ      411 Washington Street   Seward AK, 99664              Astronomy, ~
 8 KOVA      171 3rd Ave   Kotzebue AK, 99752                       Boating, Ca~
 9 LACL      1 Park Place   Port Alsworth AK, 99653                 Astronomy, ~
10 WRST      Mile 106.8 Richardson Highway  Copper Center AK, 99573 Arts and Cu~
```

Section 2

4. Build a tibble which adds 4 columns to `park_data`:

- street_address
- city
- state
- zip_code

Hint: sometimes you can extract more than you want, and then remove the extra stuff...

```
four_pieces <- park_data |>
  mutate(
    street_address = str_trim(str_extract(address, "^.*( ){2,}")),
    city = str_trim(str_extract(address, "( ){2,}.*[A-Z]{2}")),
    city = str_replace(city, " [A-Z]{2}", ""),
    state = str_extract(address, "[A-Z]{2},"),
    state = str_replace(state, ",", ""),
    zip_code = str_extract(address, "[\\d-]+$")
  ) |>
  select(address, street_address, city, state, zip_code)

four_pieces
```

```
# A tibble: 54 x 5
   address                                    street_address city  state zip_code
   <chr>                                      <chr>          <chr> <chr> <chr>
 1 25 Visitor Center Road   Bar Harbor ME, ~ 25 Visitor Ce~ Bar ~ ME    04609
 2 25216 Ben Reifel Road   Interior SD, 577~ 25216 Ben Rei~ Inte~ SD    57750
 3 1 Panther Junction   Big Bend National P~ 1 Panther Jun~ Big ~ TX    79834
 4 9700 SW 328th Street  Homestead FL, 33033 9700 SW 328th~ Home~ FL    33033
 5 9800 Highway 347  Montrose CO, 81401      9800 Highway ~ Mont~ CO    81401
 6 Highway 63 Bryce Canyon National Park  B~ Highway 63 Br~ Bryce UT    84764
 7 52 West Headquarters Drive   Torrey UT, ~ 52 West Headq~ Torr~ UT    84775
 8 727 Carlsbad Caverns Highway   Carlsbad ~ 727 Carlsbad ~ Carl~ NM    88220
 9 1901 Spinnaker Drive   Ventura CA, 93001  1901 Spinnake~ Vent~ CA    93001
10 100 National Park Road   Hopkins SC, 290~ 100 National ~ Hopk~ SC    29061
# i 44 more rows
```

[SKIP] If you had trouble producing the tibble in (4), you can read in the correct tibble using the R chunk below (use ONLY if needed!):

```
#four_pieces <- read_csv("~/264_fall_2024/DS2_preview_work/four_pieces.csv")
#four_pieces <- read_csv("~/Sds 264 F24/Class/Data/four_pieces.csv")
```

Use your new tibble from (4) to answer Questions (5) and (6).

5. Print the subset of `street_address` entries where the numerical part is 1000 or greater.

```
str_subset(four_pieces$street_address, "^\\d{4,}")
```

```
 [1] "25216 Ben Reifel Road"     "9700 SW 328th Street"
 [3] "9800 Highway 347"          "1901 Spinnaker Drive"
 [5] "6947 Riverview Road"       "40001 SR-9336"
 [7] "40001 State Road 9336"     "11999 State Highway 150"
 [9] "74485 National Park Drive" "1000 Silver Street"
[11] "38050 Highway 36 East"     "34840 Hwy 160"
[13] "55210 238th Avenue East"   "3002 Mount Angeles Road"
[15] "5000 East Entrance Road"   "1111 Second Street"
[17] "1000 US Hwy 36"            "3693 S Old Spanish Trail"
[19] "47050 Generals Highway"    "3655 U.S. Highway 211"
[21] "26611 US Highway 385"      "9039 Village Drive"
```

6. Arrange `city` names from longest to shortest.

```
four_pieces |>
  mutate(city_length = str_length(city)) |>
  select(city, city_length) |>
  arrange(desc(city_length)) |>
  print(n = Inf)
```

```
# A tibble: 54 x 2
   city                       city_length
   <chr>                            <int>
 1 Yellowstone National Park           25
 2 Big Bend National Park              22
 3 Hawaii National Park                20
 4 International Falls                 19
 5 Twentynine Palms                    16
 6 Petrified Forest                    16
 7 Port Alsworth                       13
 8 Sedro-Woolley                       13
 9 Crescent City                       13
```

10	Copper Center	13
11	Death Valley	12
12	West Glacier	12
13	Grand Canyon	12
14	Mammoth Cave	12
15	Port Angeles	12
16	Three Rivers	12
17	Crater Lake	11
18	Denali Park	11
19	Hot Springs	11
20	King Salmon	11
21	Hot Springs	11
22	Bar Harbor	10
23	Gatlinburg	10
24	Estes Park	10
25	East Luray	10
26	Springdale	10
27	Homestead	9
28	Peninsula	9
29	Homestead	9
30	Homestead	9
31	Fairbanks	9
32	Salt Flat	9
33	Interior	8
34	Montrose	8
35	Carlsbad	8
36	Gustavus	8
37	Houghton	8
38	Kotzebue	8
39	Paicines	8
40	Yosemite	8
41	Ventura	7
42	Hopkins	7
43	Mineral	7
44	Ashford	7
45	Torrey	6
46	Seward	6
47	Mancos	6
48	Tucson	6
49	Medora	6
50	Bryce	5
51	Baker	5
52	Mosca	5

```
53 Moose                          5
54 Kula                           4
```

Section 3

7. Create a new column in `park_data` which records the total number of activities in each park, then sort the parks from most activities to least.

```
park_data |>
  mutate(num_activities = str_count(activities, ",") + 1) |>
  arrange(desc(num_activities))
```

```
# A tibble: 54 x 4
   park_code address                                    activities num_activities
   <chr>     <chr>                                      <chr>               <dbl>
 1 GRSA      11999 State Highway 150  Mosca CO, 81146  Arts and ~             56
 2 GRTE      103 Headquarters Loop   Moose WY, 83012   Arts and ~             54
 3 OLYM      3002 Mount Angeles Road  Port Angeles WA~ Astronomy~             54
 4 YELL      2 Officers Row  Yellowstone National Par~ Arts and ~             53
 5 VOYA      360 Hwy 11 East  International Falls MN,~ Arts and ~             48
 6 LAVO      38050 Highway 36 East  Mineral CA, 96063  Auto and ~             47
 7 ACAD      25 Visitor Center Road   Bar Harbor ME, ~ Arts and ~             46
 8 EVER      40001 State Road 9336   Homestead FL, 33~ Auto and ~             46
 9 WRST      Mile 106.8 Richardson Highway  Copper Ce~ Arts and ~             46
10 GLAC      64 Grinnell Drive  West Glacier MT, 59936 Arts and ~             45
# i 44 more rows
```

8. Pick off all of the activities that end in "ing"; we'll refer to these as "verb activities". Produce a count of the number of parks where each "verb activity" appears, and print the "verb activities" and their counts in order from most parks to fewest. (Note that you should consider something like "Group Camping" as different from "RV Camping" or just plain "Camping".) Your answer should look like the tibble below:

```
# A tibble: 57 × 2
   verb_activity              n
   <chr>                  <int>
 1 Hiking                    50
 2 Shopping                  46
 3 Stargazing                34
 4 Wildlife Watching         31
 5 Camping                   30
```

```
 6 Scenic Driving            26
 7 Horse Trekking            23
 8 Canoe or Kayak Camping    22
 9 Group Camping             22
10 Paddling                  21
#   47 more rows```
```

Hint: if you produce a list where each element in the list is a vector (with differing numbers of strings), you can use `unlist` to produce a single character vector

```
#Approach 1
#str_view(park_data$activities, "([A-Z][a-z\\- ]*)* [A-Z][a-z]+ing\\b")

activities_vector <- unlist(str_extract_all(park_data$activities, "([A-Z][a-z\\- ]*)* [A-Z][a

verb_activity2 <- unlist(str_extract_all(park_data$activities,
                        "([A-Z][a-z\\- ]*)* [A-Z][a-z]+ing\\b"))
verb_activity2 <- str_remove(verb_activity2, "^ ")
park_verbs2 <- tibble(verb_activity = verb_activity2) |>
  count(verb_activity) |>
  arrange(desc(n))

#print(park_verbs2, n = Inf)


#Approach 2
#str_view(park_data$activities, "\\b[A-Za-z\\s\\-\\(\\)\\/]+ing\\b")

list_verbs <- str_extract_all(park_data$activities, "\\b[A-Za-z\\s\\-\\(\\)\\/]+ing\\b")

verbs <- as.tibble((unlist(list_verbs))) |>
  mutate(verb_activity = value) |>
  count(verb_activity) |>
  arrange(desc(n))
```

```
Warning: `as.tibble()` was deprecated in tibble 2.0.0.
i Please use `as_tibble()` instead.
i The signature and semantics have changed, see `?as_tibble`.
```

```
#print(verbs, n = Inf)
```

```r
# Final approach
activities <- str_trim(unlist(str_split(park_data$activities, ",")))
park_verbs <- tibble(verb_activity = activities) |>
  mutate(verb_activity = str_trim(
    str_replace(verb_activity, "\\(.*\\)", ""))) |>
  filter(str_detect(verb_activity, "ing$")) |>
  count(verb_activity) |>
  arrange(desc(n))

#print(park_verbs, n = Inf)

# Biggest issue: Horse Camping (see also camping) and Horse Camping
#   (see also Horse/Stock Use) get both counted as Horse Camping,
#   and if a park has both, Horse Camping gets counted twice.

# Comparisons:
temp1 <- full_join(park_verbs, park_verbs2,
                   by = c("verb_activity"="verb_activity"))
temp2 <- full_join(temp1, verbs,
                   by = c("verb_activity"="verb_activity"))
print(temp2, n = Inf)
```

```
# A tibble: 74 x 4
   verb_activity                    n.x   n.y     n
   <chr>                          <int> <int> <int>
 1 Camping                           53    50    53
 2 Hiking                            52    52    52
 3 Shopping                          51    51    51
 4 Wildlife Watching                 48    48    48
 5 Backcountry Camping               46    46    46
 6 Birdwatching                      43    43    43
 7 Backcountry Hiking                39    39    39
 8 Front-Country Hiking              39    39    39
 9 Biking                            38    38    38
10 Fishing                           37    37    37
11 Car or Front Country Camping      36    36    36
12 Horse Camping                     34    34    20
13 Stargazing                        34    34    34
14 Canoe or Kayak Camping            31    31    31
15 Group Camping                     29    29    29
16 Picnicking                        29    29    29
```

17	Paddling	28	28	28
18	Horse Trekking	27	27	27
19	Scenic Driving	26	26	26
20	RV Camping	25	25	25
21	Boating	23	20	23
22	Climbing	23	23	23
23	Road Biking	22	22	22
24	Snowshoeing	22	22	22
25	Horseback Riding	21	21	21
26	Skiing	21	21	21
27	Kayaking	20	20	20
28	Cross-Country Skiing	19	19	19
29	Dining	19	19	19
30	Freshwater Fishing	19	19	19
31	Self-Guided Tours - Walking	19	19	19
32	Canoeing	17	17	17
33	Fly Fishing	14	14	14
34	Swimming	14	14	14
35	Off-Trail Permitted Hiking	13	13	13
36	Mountain Climbing	11	11	11
37	Rock Climbing	11	11	11
38	Flying	10	10	10
39	Hunting and Gathering	10	NA	10
40	Motorized Boating	10	10	10
41	Snowmobiling	9	9	9
42	Stand Up Paddleboarding	9	9	9
43	Hunting	8	18	8
44	Saltwater Fishing	7	7	7
45	Caving	6	6	6
46	Mountain Biking	6	6	6
47	Sailing	6	6	6
48	Whitewater Rafting	6	6	6
49	Freshwater Swimming	5	5	5
50	Geocaching	5	5	5
51	Orienteering	5	5	5
52	Saltwater Swimming	5	5	5
53	Canyoneering	4	4	4
54	Fixed Wing Flying	4	4	4
55	Gathering and Foraging	4	NA	4
56	SCUBA Diving	4	4	4
57	Tubing	4	4	4
58	Auto Off-Roading	3	NA	3
59	Downhill Skiing	3	3	3

```
60 Snorkeling                          3     3     3
61 Snow Tubing                         3     3     3
62 Surfing                             3     3     3
63 Water Skiing                        3     3     3
64 ATV Off-Roading                     2    NA     2
65 Dog Sledding                        2     2     2
66 Ice Climbing                        2     2     2
67 Ice Skating                         2     2     2
68 Helicopter Flying                   1     1     1
69 Pool Swimming                       1     1     1
70 River Tubing                        1     1     1
71 Gathering                          NA    14    NA
72 Living                             NA     7     7
73 Foraging                           NA     4    NA
74 Horse Camping (see also camping    NA    NA    14
```

Use your tibble from (8) to answer Questions (9)-(10).

[SKIP] If you had trouble producing the tibble in (8), you can read in the correct tibble using the R chunk below (use ONLY if needed!):

```
#active_counts <- read_csv("~/264_fall_2024/DS2_preview_work/active_counts.csv")
#active_counts <- read_csv("~/Sds 264 F24/Class/Data/active_counts.csv")
```

9. Print all the "verb activities" that have a capital letter / lower case letter combination that repeats later in the phrase (e.g. "Gh" appears twice).

```
str_subset(park_verbs$verb_activity, "([A-Z][a-z]).*\\1")
```

```
[1] "Car or Front Country Camping" "Canoe or Kayak Camping"
```

10. Print all the "verb activities" that have the same consonant appear twice in a row.

```
str_subset(str_to_lower(park_verbs$verb_activity), "([^aeiou])\\1")
```

```
 [1] "shopping"                  "paddling"
 [3] "horse trekking"            "cross-country skiing"
 [5] "swimming"                  "off-trail permitted hiking"
 [7] "stand up paddleboarding"   "freshwater swimming"
 [9] "saltwater swimming"        "auto off-roading"
[11] "downhill skiing"           "atv off-roading"
[13] "dog sledding"              "pool swimming"
```