Factors and Strings

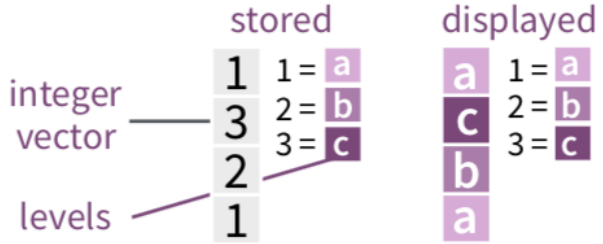
Fall 2022

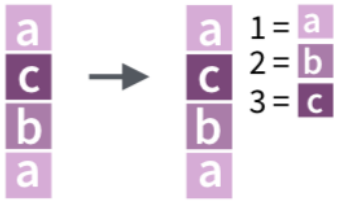
October 09 2022

Factors - categorical data

Factors

R represents categorical data with factors. A **factor** is an integer vector with a **levels** attribute that stores a set of mappings between integers and categorical values. When you view a factor, R displays not the integers, but the values associated with them.

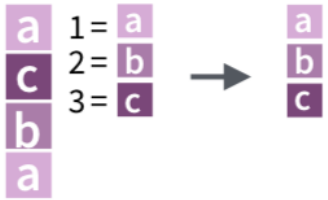




Create a factor with factor()

factor(x = character(), levels, labels = levels, exclude = NA, ordered = is.ordered(x), nmax = NA) Convert a vector to a factor. Also **as_factor**.

```
f <- factor(c("a", "c", "b", "a"),
            levels = c("a", "b", "c"))
```



Return its levels with levels()

levels(x) Return/set the levels of a factor. *levels(f); levels(f) <- c("x","y","z")*

Use unclass() to see its structure

- Clean and order factors with `forcats` package
- Important for visualization, statistical modeling (i.e. for `lm()`), and creating tables

See `forcats` cheatsheet and `forcats` vignette

Example - specify levels `fct_relevel()`

```
mydata <- tibble(
  id = 1:4,
  grade=c("9th","10th","11th","9th")) %>%
  mutate(grade_fac = factor(grade))
levels(mydata$grade_fac)
[1] "10th" "11th" "9th"
```

```
mydata %>%
  arrange(grade_fac)
# A tibble: 4 × 3
     id grade grade_fac
  <int> <chr> <fct>
1     2 10th  10th
2     3 11th  11th
3     1 9th   9th
4     4 9th   9th
```

```
mydata <- mydata %>%
  mutate(
    grade_fac =
      fct_relevel(grade_fac,
                  c("9th","10th","11th")))
levels(mydata$grade_fac)
[1] "9th"  "10th" "11th"
mydata %>% arrange(grade_fac)
# A tibble: 4 × 3
     id grade grade_fac
  <int> <chr> <fct>
1     1 9th   9th
2     4 9th   9th
3     2 10th  10th
4     3 11th  11th
```

Example - collapse levels `fct_collapse()`

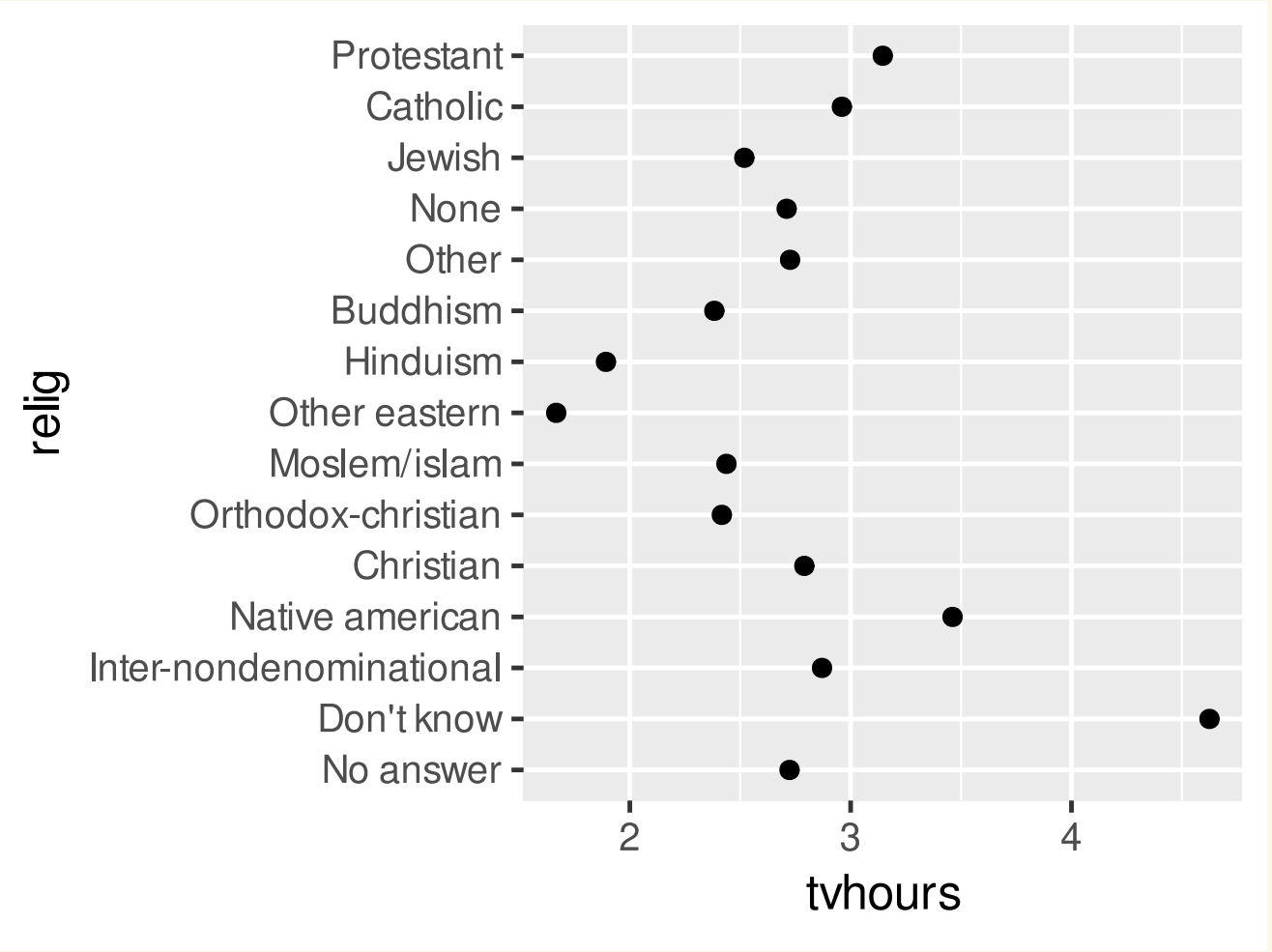
```
mydata <- tibble(loc = c("SW","NW","NW","NE","SE","SE"))
mydata %>% mutate(
  loc_fac = factor(loc),
  loc2 = fct_collapse(loc_fac,                          # collapse levels
                      south = c("SW","SE"),
                      north = c("NE","NW")),
  loc3 = fct_lump(loc_fac, n=2, other_level = "other") # most common 2 levels + other
  )
# A tibble: 6 × 4
  loc   loc_fac loc2  loc3
  <chr> <fct>   <fct> <fct>
1 SW    SW      south other
2 NW    NW      north NW
3 NW    NW      north NW
4 NE    NE      north other
5 SE    SE      south SE
6 SE    SE      south SE
```

Example - collapse levels `fct_collapse()`

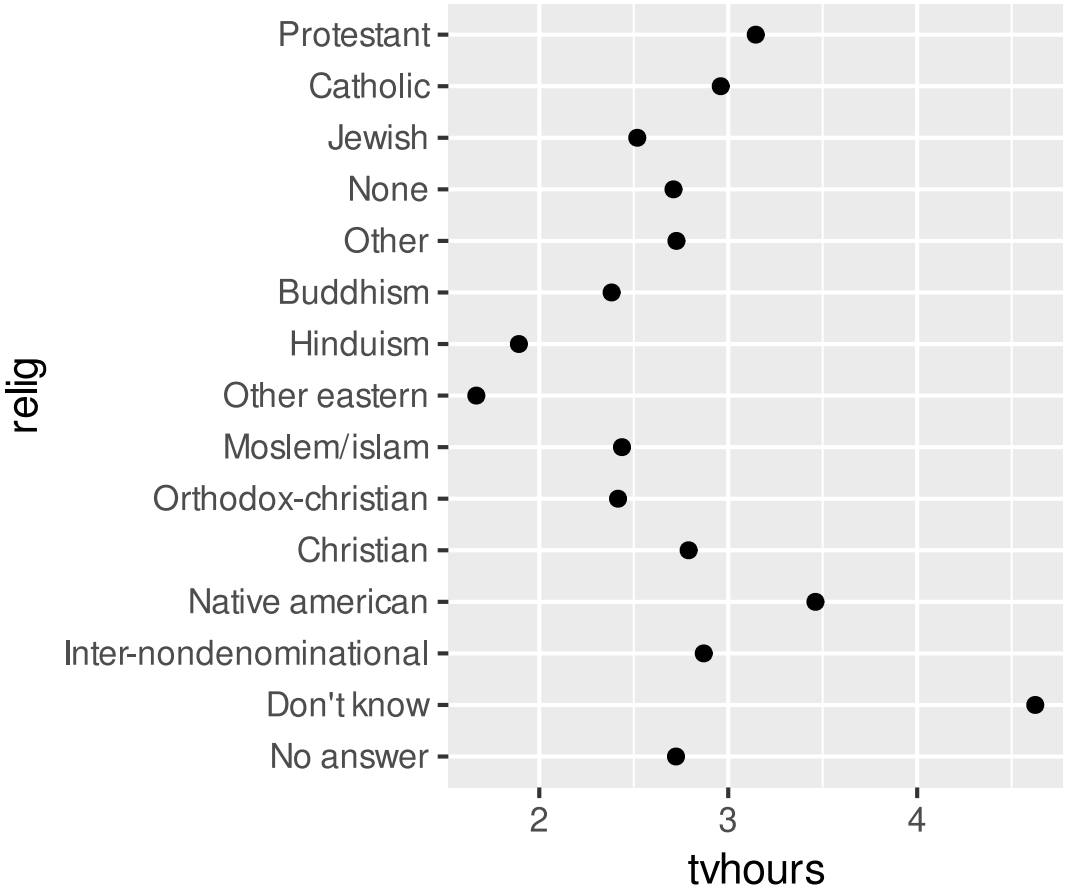
```
mydata <- tibble(loc = c("SW","NW","NW","NE","SE","SE"))
mydata %>% mutate(
  loc_fac = factor(loc),
  loc2 = fct_collapse(loc_fac,                          # collapse levels
                      south = c("SW","SE"),
                      north = c("NE","NW")),
  loc3 = fct_lump(loc_fac,
                  n=2,
                  other_level = "other") # most common 2 levels + other
  )
# A tibble: 6 × 4
  loc   loc_fac loc2  loc3
  <chr> <fct>   <fct> <fct>
1 SW    SW      south other
2 NW    NW      north NW
3 NW    NW      north NW
4 NE    NE      north other
5 SE    SE      south SE
6 SE    SE      south SE
```

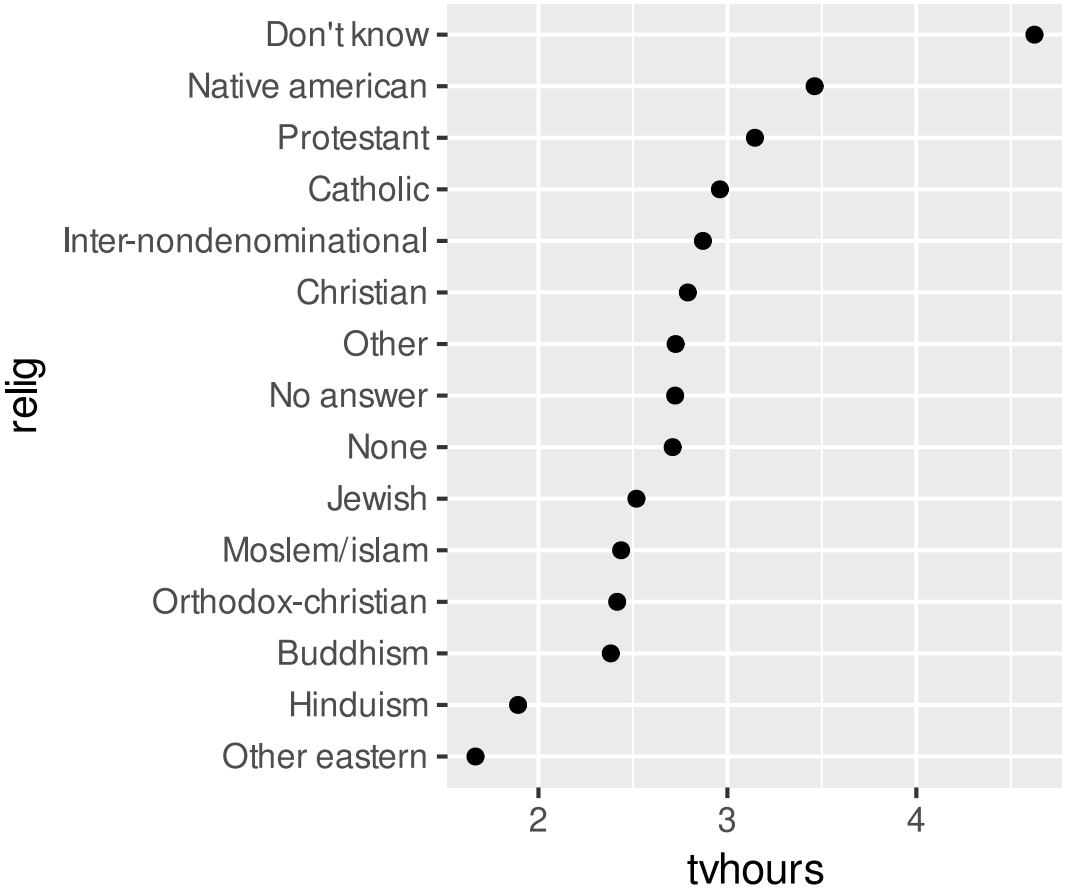
Which religions watch the least TV?

```
gss_cat %>%
  drop_na(tvhours) %>%
  group_by(relig) %>%
  summarize(tvhours = mean(tvhours))
  ggplot(aes(tvhours, relig)) +
    geom_point()
```



Which one do you prefer?





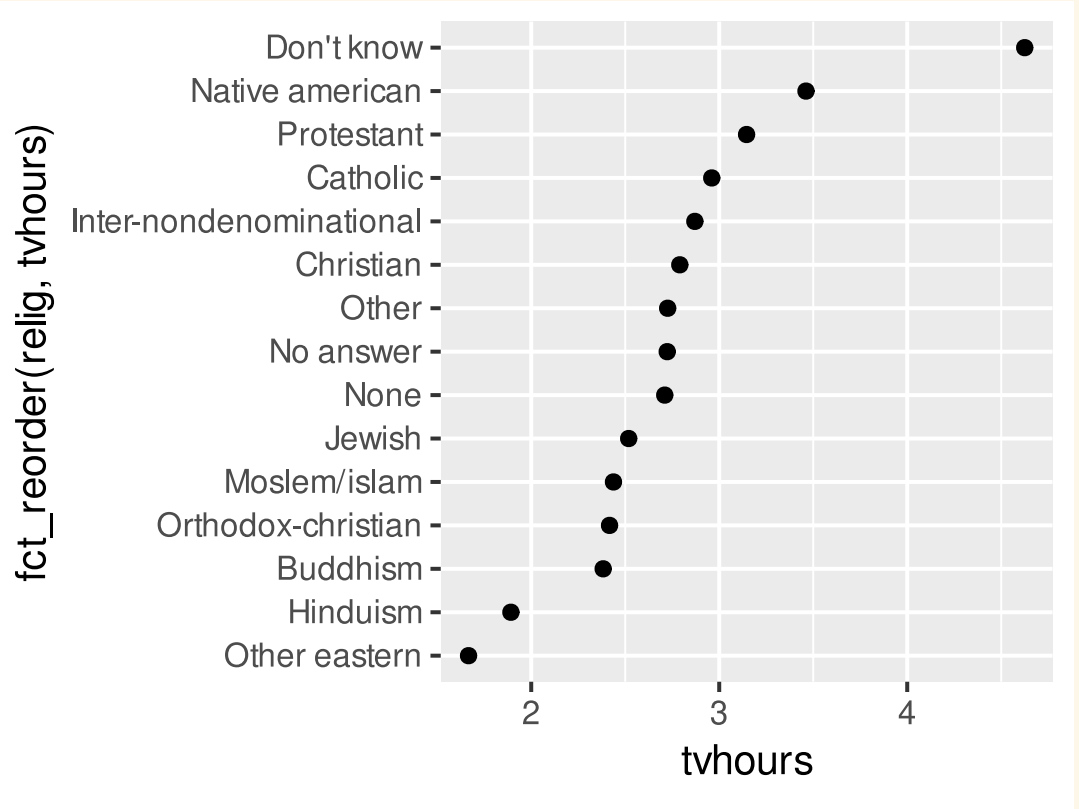
Use levels() to access a factor's levels

```
gss_cat %>%
  pull(relig) %>%
  levels() %>%
  kable()
```

x
No answer
Don't know
Inter-nondenominational
Native american
Christian
Orthodox-christian
Moslem/islam
Other eastern
Hinduism
Buddhism
Other
None
Jewish
Catholic
Protestant
Not applicable

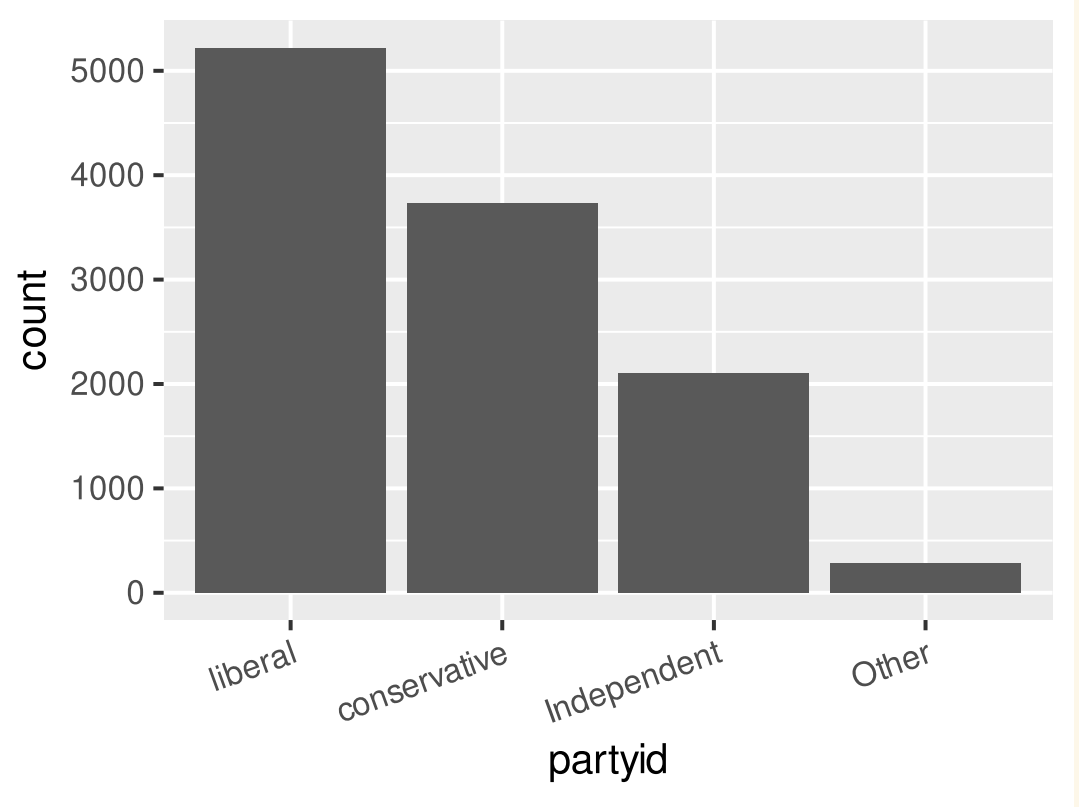
Reorder relig by tvhours

```
gss_cat %>%
  drop_na(tvhours) %>%
  group_by(relig) %>%
  summarize(tvhours = mean(tvhours)) %>%
  ggplot(aes(
    x = tvhours,
    y = fct_reorder(relig, tvhours)
  )) +
    geom_point()
```



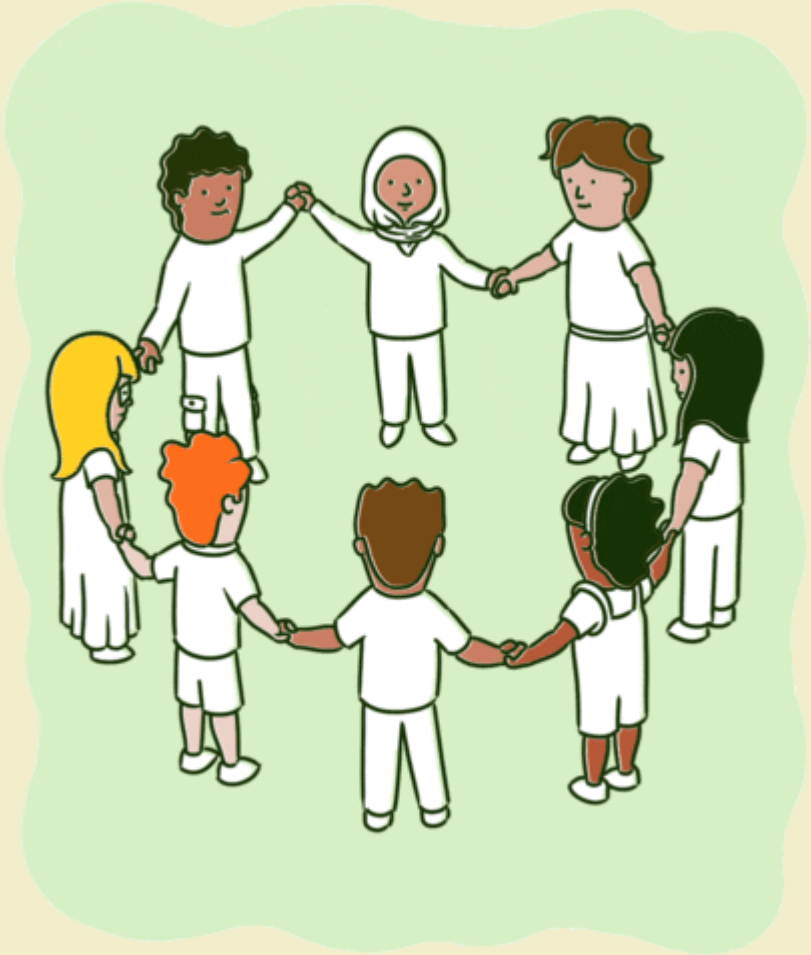
Lumping partyid: fct_lump()

```
gss_cat %>%
  mutate(partyid = fct_lump(partyid, n = 3)) %>%
  ggplot(aes(x = fct_infreq(partyid))) +
  geom_bar() +
  theme(axis.text.x = element_text(angle = 20,
                                   vjust = 1,
                                   hjust=1)) +
  labs(x = "partyid")
```



Group Activity 1

10:00



- Let's go over to maize server/ local Rstudio and our class moodle
- Get the class activity 12.Rmd file
- Work on problem 1
- Ask me questions

Let's do some more string manipulation!!

Last time: Quantifiers and Special Characters

Preceding characters are matched … .

- `*` = 0 or more
- `?` = 0 or 1
- `+` = 1 or more
- `{n}` = exactly n times

Matching character types

- `\\d` = digit
- `\\s` = white space
- `\\w` = word
- `\\t` = tab
- `\\n` = newline

More quantifiers

useful when you want to match a pattern a specific number of times

- `{n, }` = n or more times
- `{, m}` = at most m times
- `{n, m}` = between n & m times

Alternatives

useful for matching patterns more flexibly

- `[abc]` = one of a, b, or c
- `[e-z]` = a letter from e to z
- `[^abc]` = anything other than a, b, or c

str_view_all()

```
name_phone <- c("Moly Robins: 250-999-8878",
        "Ali Duluth: 416-908-2044",
        "Eli Mitchell: 204.192.9829",
        "May Flowers: 250.209.7047")
```

```
str_view_all(name_phone,
             pattern = "([2-9][0-9]{2})[.-]([0-9]{3})[.-]([0-9]{4})")
```

Moly Robins: 250-999-8878

Ali Duluth: 416-908-2044

Eli Mitchell: 204.192.9829

May Flowers: 250.209.7047

Replacing strings

```
str_replace_all(name_phone,
pattern = "([2-9][0-9]{2})[.-]([0-9]{3})[.-]([0-9]{4})",
replacement = "XXX-XXX-XXXX"
)
[1] "Moly Robins: XXX-XXX-XXXX"  "Ali Duluth: XXX-XXX-XXXX"
[3] "Eli Mitchell: XXX-XXX-XXXX" "May Flowers: XXX-XXX-XXXX"
```

str_extract_all()

pull all set of values matching the specified pattern

```
name_phone <- c("Moly Robins: 250-999-8878",
                "Ali Duluth: 416-908-2044",
                "Eli Mitchell: 204-192-9829",
                "May Flowers: 250-209-7047")
```

```
str_extract_all(name_phone, "[:alpha:]{2,}", simplify = TRUE)
     [,1]   [,2]
[1,] "Moly" "Robins"
[2,] "Ali"  "Duluth"
[3,] "Eli"  "Mitchell"
[4,] "May"  "Flowers"
```

Duplicating Groups

Use escaped numbers (\\1, \\2, etc) to repeat a group based on position

Which numbers have the same 1st and 3rd digits?

```
phone_numbers <- c("515 111 2244", "310 549 6892", "474 234 7548")
str_view(phone_numbers, "(\\d)\\d\\1")
```

```
515 111 2244
310 549 6892
474 234 7548
```

Repetition using ?

```
aboutMe <- c("my SSN is 536-76-9423 and my age is 55")
```

```
str_view_all(aboutMe, "\\s\\d?") # space followed by 0 or 1 digit
```

```
my SSN is 536-76-9423 and my age is 55
```

Repetition using +

```
aboutMe <- c("my SSN is 536-76-9423 and my age is 55")
```

```
str_view_all(aboutMe, "\\s\\d+")  # space followed by 1 or more digi
```

my SSN is 536-76-9423 and my age is 55

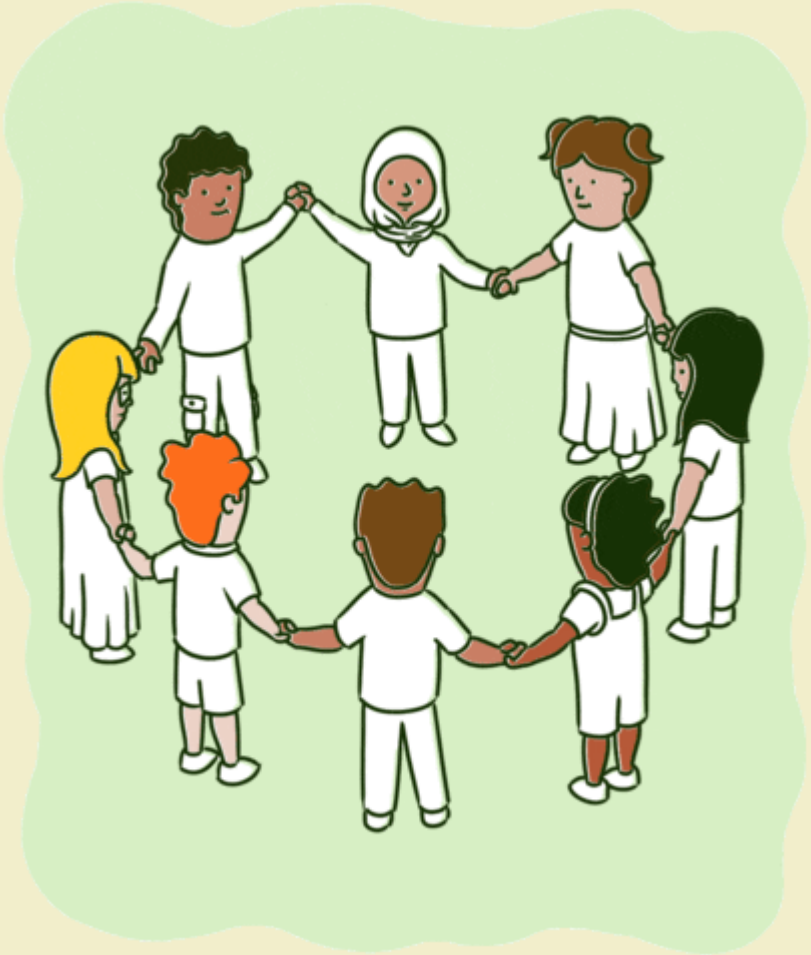
Repetition using *

```
aboutMe <- c("my SSN is 536-76-9423 and my age is 55")
```

```
str_view_all(aboutMe, "\\s\\d*")  # space followed by 0 or more digi
```

Group Activity 2

10:00



- Go back to the activity file
- Work on problem 2
- Ask me questions