HW 13

Due Tues April 18

Al Ashir Intisar

This homework includes some basics of STRINGS and FUNCTIONS

```
library(tidyverse)
```

Part 1: mini_tibble

```
mini_tibble <- tibble(produce = c("apple", "poblano", "banana"),
                      class_num = c(110, 212, 272))

mini_tibble
```

```
## # A tibble: 3 x 2
##   produce class_num
##   <chr>       <dbl>
## 1 apple         110
## 2 poblano       212
## 3 banana        272
```

For the following problems, you can just print out the resulting dataset, you don't need to save the output.

1. Add a column to mini_tibble called "last" which is the last character of the produce variable. (hint: str_sub).

```
mini_tibble|>
  mutate(last = str_sub(produce, start = str_length(produce)))
```

```
## # A tibble: 3 x 3
##   produce class_num last
##   <chr>       <dbl> <chr>
## 1 apple         110 e
## 2 poblano       212 o
## 3 banana        272 a
```

2. Add a column to mini_tibble called "stats_classes" which has values "STAT 110", "STAT 212", "STAT 272". (hint: str_c)

```
mini_tibble|>
  mutate(stats_classes = str_c("STAT ", class_num))
```

```
## # A tibble: 3 x 3
##   produce class_num stats_classes
##   <chr>       <dbl> <chr>
## 1 apple         110 STAT 110
## 2 poblano       212 STAT 212
## 3 banana        272 STAT 272
```

3. [For this problem, don't use mini_tibble!] Use `str_length()` and `str_sub()` to extract the **middle** character from the string "apple". Then use the same two functions to extract the middle character from the string "poblano". Finally, use the same two functions to extract the middle two characters from the string "banana".

The code below gets you started.

```
x <- "apple"
middle <- (str_length(x) + 1) / 2
str_sub(x, middle, middle)
```

```
## [1] "p"
```

```
x <- "poblano"
middle <- (str_length(x) + 1) / 2
str_sub(x, middle, middle)
```

```
## [1] "l"
```

```
x <- "banana"
middle <- (str_length(x) + 1) / 2
str_sub(x, middle, middle+1)
```

```
## [1] "na"
```

4. [For this problem, don't use mini_tibble!] Write a function (call it `extract_middle()`) that will extract the middle character from a string with an odd number of charcters and the middle two characters from a string with an even number of characters. Test it on "apple", "poblano", and "banana". (Hint: x %% 2 == 0 is TRUE if x is even and FALSE if x is odd)

```
extract_middle <- function(x){

  middle <- (str_length(x) + 1) / 2

  ifelse(str_length(x) %% 2 == 0, str_sub(x, middle, middle+1), str_sub(x, middle, middle))


}


extract_middle("apple")
```

```
## [1] "p"
```

```
extract_middle("poblano")
```

```
## [1] "l"
```

```
extract_middle("banana")
```

```
## [1] "na"
```

5. Show how you can use your new function to add a column to mini_tibble called "middle" which has the middle 1 or 2 characters.

```
mini_tibble|>
  mutate(middle = extract_middle(produce))
```

```
## # A tibble: 3 x 3
##   produce class_num middle
##   <chr>       <dbl> <chr>
## 1 apple         110 p
## 2 poblano       212 l
## 3 banana        272 na
```

Part 2: Spotify

```
bigspotify <- read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2020/
```

```
## Rows: 32833 Columns: 23
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): track_id, track_name, track_artist, track_album_id, track_album_na...
## dbl (13): track_popularity, danceability, energy, key, loudness, mode, speec...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
spot_smaller <- bigspotify[c(3993,  1590, 23036, 23062, 18304, 20630,  6193,  7922, 21105,  9432), ] %>%
  select(track_name, track_artist, track_album_release_date, playlist_genre, playlist_subgenre, playlis

spot_smaller
```

```
## # A tibble: 10 x 6
##    track_name                            track~1 track~2 playl~3 playl~4 playl~5
##    <chr>                                 <chr>   <chr>   <chr>   <chr>   <chr>
##  1 Hear Me Now                           Alok    2016-0~ pop     indie ~ Chillo~
##  2 Run the World (Girls)                 Beyoncé 2011-0~ pop     post-t~ post-t~
##  3 Formation                             Beyoncé 2016-0~ r&b     hip pop Feelin~
##  4 7/11                                  Beyoncé 2014-1~ r&b     hip pop Feelin~
##  5 My Oh My (feat. DaBaby)               Camila~ 2019-1~ latin   latin ~ 2020 H~
##  6 It's Automatic                        Freest~ 2013-1~ latin   latin ~ 80's F~
##  7 Poetic Justice                        Kendri~ 2012    rap     hip hop Hip Ho~
##  8 A.D.H.D                               Kendri~ 2011-0~ rap     southe~ Hip-Ho~
##  9 Ya Estuvo                             Kid Fr~ 1990-0~ latin   latin ~ HIP-HO~
## 10 Runnin (with A$AP Rocky, A$AP Ferg &~ Mike W~ 2018-1~ rap     gangst~ RAP Ga~
## # ... with abbreviated variable names 1: track_artist,
## #   2: track_album_release_date, 3: playlist_genre, 4: playlist_subgenre,
## #   5: playlist_name
```

6. Use $ and [] to print the first three track names from spot_smaller. (Hint: See Code > 06-functions_Vectors_classes > Ch20_tibbles Rmd)

```
spot_smaller$track_name[1:3]
```

```
## [1] "Hear Me Now"            "Run the World (Girls)" "Formation"
```

7. The file Class > 07-Strings > ch14_str_functions.Rmd includes examples of how to view and detect certain strings, and how to use str_detect within a filter statement. Modify the code below to find songs with a track_name that contains the word "Run"

```
# These will not knit because Html previewer cannot knit.
#str_view(spot_smaller$playlist_subgenre, "pop")
#str_view(spot_smaller$playlist_subgenre, "pop", match = TRUE)
```

```
str_detect(spot_smaller$playlist_subgenre, "pop")
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
```

```
spot_smaller %>%
  filter(str_detect(track_name, "Run "))
```

```
## # A tibble: 1 x 6
##   track_name            track_artist track_album_relea~1 playl~2 playl~3 playl~4
##   <chr>                 <chr>        <chr>               <chr>   <chr>   <chr>
## 1 Run the World (Girls) Beyoncé      2011-06-24          pop     post-t~ post-t~
## # ... with abbreviated variable names 1: track_album_release_date,
## #   2: playlist_genre, 3: playlist_subgenre, 4: playlist_name
```

8. Using the last filter, what happens if you look for the string "run" in track_name instead? Add a mutate step so that the track_name is converted to all lower case before filtering to find "run".

```
spot_smaller %>%
  filter(str_detect(track_name, "run"))
```

```
## # A tibble: 0 x 6
## # ... with 6 variables: track_name <chr>, track_artist <chr>,
## #   track_album_release_date <chr>, playlist_genre <chr>,
## #   playlist_subgenre <chr>, playlist_name <chr>
## # i Use `colnames()` to see all variable names
```

```
spot_smaller|>
  mutate(track_name = str_to_lower(track_name))|>
  filter(str_detect(track_name, "run"))
```

```
## # A tibble: 2 x 6
##   track_name                              track~1 track~2 playl~3 playl~4 playl~5
##   <chr>                                   <chr>   <chr>   <chr>   <chr>   <chr>
## 1 run the world (girls)                   Beyoncé 2011-0~ pop     post-t~ post-t~
## 2 runnin (with a$ap rocky, a$ap ferg & ~ Mike W~ 2018-1~ rap     gangst~ RAP Ga~
## # ... with abbreviated variable names 1: track_artist,
## #   2: track_album_release_date, 3: playlist_genre, 4: playlist_subgenre,
## #   5: playlist_name
```

Ans: When used "run" we don't find any track_name with that word because lower case r is different than upper case R.

9. Repeat number 8 with the entire bigspotify dataset. How many songs contain "run" in the track_name?

```
bigspotify|>
  mutate(track_name = str_to_lower(track_name))|>
  mutate(run_num = str_detect(track_name, "run"))|>
  drop_na(run_num)|>
  summarise(run = sum(run_num))
```

```
## # A tibble: 1 x 1
##     run
##   <int>
## 1   182
```

Ans: There are 182 songs that contain "run" in the track_name.