Modifying Strings

Step 1: Load and process your data.

Load the calls data set, and process it to split the address, town, and date-time variables:

```
library(tidyverse)
calls <- read.csv("calls.csv", stringsAsFactors = FALSE)

# Initialize vectors to store each of the new variables
address <- c()
town <- c()
dt <- c()

for(i in 1:nrow(calls)) { # loop over emergency calls

  # get the description of the i^th call
  callI <- calls[i, "desc"]

  # split the description text based on ";" --> gives a matrix of substrings
  splitCallDesc <- str_split(callI, ";", simplify = TRUE)

  # store the street address, town, and date/time
  address[i] <- splitCallDesc[1]
  town[i] <- splitCallDesc[2]
  dt[i] <- splitCallDesc[3]

}

# add the new variables to the data frame
calls$address <- address
calls$towns <- town
calls$dt <- dt
```

Step 2: Replace "&" with "AND".

Take a look at the first few rows of the address data. Notice that some intersections use "&" and others use "AND":

```
head(calls$address)
```

```
## [1] "BRADFIELD RD & SUSQUEHANNA RD"  "E CITY AVE & PRESIDENTIAL BLVD"
## [3] "MAPLE AVE AND W 6TH ST"         "DEKALB ST"
## [5] "BEECH ST"                       "DEKALB ST AND W 5TH ST"
```

To replace all instances of & with AND, use the code:

```
calls$address <- str_replace(calls$address, " AND ", " & ")
head(calls$address)
```

```
## [1] "BRADFIELD RD & SUSQUEHANNA RD"  "E CITY AVE & PRESIDENTIAL BLVD"
```

```
## [3] "MAPLE AVE & W 6TH ST"           "DEKALB ST"
## [5] "BEECH ST"                       "DEKALB ST & W 5TH ST"
```

Note that the spaces here are important because they keep R from replacing parts of words that include the string "AND".

Step 3: Remove excess white space.

Splitting the strings on the semicolon introduces extra white space that you should remove. For example, if you look at the towns, you can see that some are surrounded by more space than others:

```
head(calls$towns)
```

```
## [1] " ABINGTON"     " LOWER MERION" "  LANSDALE"    " BRIDGEPORT"
## [5] "  POTTSTOWN"   " BRIDGEPORT"
```

Use str_trim() to remove the excess white space:

```
calls$towns <- str_trim(calls$towns)
head(calls$towns)
```

```
## [1] "ABINGTON"     "LOWER MERION" "LANSDALE"     "BRIDGEPORT"   "POTTSTOWN"
## [6] "BRIDGEPORT"
```

Step 4: Change capitalization within strings.

Make all the capitalization in a string consistent by using the commands str_to_upper() or str_to_lower(). Both of these functions take a vector of strings as their argument:

```
calls$title <- str_to_upper(calls$title)
head(calls$title)
```

```
## [1] "FIRE: FIRE ALARM"          "TRAFFIC: VEHICLE ACCIDENT"
## [3] "EMS: LACERATIONS"          "FIRE: WOODS/FIELD FIRE"
## [5] "EMS: STABBING"             "TRAFFIC: VEHICLE ACCIDENT"
```