Assignment 4

Landon Wilson

2024-10-25

```
library(tidyverse)
library(stringr)
library(refinr)
```

Exercise 1

For the following regular expression, explain in words what it matches on. Then add test strings to demonstrate that it in fact does match on the pattern you claim it does. Make sure that your test set of strings has several examples that match as well as several that do not. Show at least two examples that return TRUE and two examples that return FALSE. *If you copy the Rmarkdown code for these exercises directly from my source pages, make sure to remove the* `eval=FALSE` *from the R-chunk headers.*

Here is an example of what a solution might look like.

q) This regular expression matches:

Any string that contains the lower-case letter "a".

```
strings <- c('Adel', 'Mathematics', 'able', 'cheese')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, 'a') )
```

```
##        string result
## 1        Adel  FALSE
## 2 Mathematics   TRUE
## 3        able   TRUE
## 4      cheese  FALSE
```

Please complete the questions below.

a) This regular expression matches:

Matches any string containing lowercase "ab"

```
strings <- c("chair","table","letters","abc")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, 'ab') )
```

```
##    string result
## 1   chair  FALSE
## 2   table   TRUE
## 3 letters  FALSE
## 4     abc   TRUE
```

b) This regular expression matches:

Matches any string containing lowercase a or b or both

```
strings <- c("golem","fail","win","bloated")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '[ab]') )
```

```
##    string result
## 1   golem  FALSE
## 2    fail   TRUE
## 3     win  FALSE
## 4 bloated   TRUE
```

c) This regular expression matches:

Matches a string starting with a lowercase a or b

```
strings <- c("soda","apple","drink","blueberry")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '^[ab]') )
```

```
##      string result
## 1      soda  FALSE
## 2     apple   TRUE
## 3     drink  FALSE
## 4 blueberry   TRUE
```

d) This regular expression matches:

Matches a number followed by a space and an uppercase or lowercase A

```
strings <- c("9 forest","6 apples","11 teens","15 Andersons")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '\\d+\\s[aA]') )
```

```
##         string result
## 1     9 forest  FALSE
## 2     6 apples   TRUE
## 3     11 teens  FALSE
## 4 15 Andersons   TRUE
```

e) This regular expression matches:

Matches a number followed by an optional space and an uppercase or lowercase A

```
strings <- c("9forest","6apples","11 teens","15 Andersons")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '\\d+\\s*[aA]') )
```

```
##         string result
## 1      9forest  FALSE
## 2      6apples   TRUE
## 3     11 teens  FALSE
## 4 15 Andersons   TRUE
```

f) This regular expression matches:

Matches any strings

```
strings <- c("sring","6 apples","17","15 Andersons")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '.*') )
```

```
##         string result
## 1        sring   TRUE
## 2     6 apples   TRUE
## 3           17   TRUE
## 4 15 Andersons   TRUE
```

g) This regular expression matches:

Matches strings that have 2 characters and "bar" following it

```
strings <- c("footballbar", "12bars in the city", "911bar", "xxbars")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '^\\w{2}bar') )
```

```
##               string result
## 1        footballbar  FALSE
## 2 12bars in the city   TRUE
## 3             911bar  FALSE
## 4             xxbars   TRUE
```

h) This regular expression matches:

Matches strings with "foo.bar" OR 2 characters and "bar" following it(previous one)

```
strings <- c("football bar", "12bars in the city", "911bar", "xxbars foo.bar")
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '(foo\\.bar)|(^\\w{2}bar)') )
```

```
##               string result
## 1       football bar  FALSE
## 2 12bars in the city   TRUE
## 3             911bar  FALSE
## 4     xxbars foo.bar   TRUE
```

Exercise 2

The following file names were used in a camera trap study. The S number represents the site, P is the plot within a site, C is the camera number within the plot, the first string of numbers is the YearMonthDay and the second string of numbers is the HourMinuteSecond.

```
file.names <- c( 'S123.P2.C10_20120621_213422.jpg',
                 'S10.P1.C1_20120622_050148.jpg',
                 'S187.P2.C2_20120702_023501.jpg')
```

Produce a data frame with columns corresponding to the `site`, `plot`, `camera`, `year`, `month`, `day`, `hour`, `minute`, and `second` for these three file names. So we want to produce code that will create the data frame:

```
 Site Plot Camera Year Month Day Hour Minute Second
 S123   P2    C10 2012    06  21   21     34     22
  S10   P1     C1 2012    06  22   05     01     48
 S187   P2     C2 2012    07  02   02     35     01
```

```
Big.sections <- str_split_fixed(file.names, pattern = "_|\\.",n = 6) #breaking up sections by _ or .
year.sec <-substr(Big.sections[,4],1,4) #selecting character 1-4 from section 4
months.sec <- substr(Big.sections[,4],5,6) #selecting character 5-6 from section 4
day.sec <-substr(Big.sections[,4],7,8)#selecting character 7-8 from section 4
hour.sec <-substr(Big.sections[,5],1,2)#selecting character 1-2 from section 5
min.sec <-substr(Big.sections[,5],3,4)#selecting character 3-4 from section 5
sec.sec <-substr(Big.sections[,5],5,6)#selecting character 5-6 from section 5

data.frame( Site=Big.sections[,1], # selects the first main section
            Plot=Big.sections[,2], # selects the second main section
            Camera=Big.sections[,3],# selects the fourth main section
            Year=year.sec,
            Month=months.sec,
            Day=day.sec,
            Hour=hour.sec,
            Minute=min.sec,
            Seconds=sec.sec)
```

```
##   Site Plot Camera Year Month Day Hour Minute Seconds
## 1 S123   P2    C10 2012    06  21   21     34      22
## 2  S10   P1     C1 2012    06  22   05     01      48
## 3 S187   P2     C2 2012    07  02   02     35      01
```

Exercise 3

The full text from Lincoln's Gettysburg Address is given below. It has been provided in a form that includes lots of different types of white space. Your goal is to calculate the mean word length of Lincoln's Gettysburg Address! *Note: you may consider 'battle-field' as one word with 11 letters or as two words 'battle' and 'field'. The first option a bit more difficult and technical!.*

```
Gettysburg <- 'Four score and seven years ago our fathers brought forth on this
continent, a new nation, conceived in Liberty, and dedicated to the proposition
that all men are created equal.

Now we are engaged in a great civil war, testing whether that nation, or any
nation so conceived and so dedicated, can long endure. We are met on a great
battle-field of that war. We have come to dedicate a portion of that field, as
a final resting place for those who here gave their lives that that nation might
live. It is altogether fitting and proper that we should do this.

But, in a larger sense, we can not dedicate -- we can not consecrate -- we can
not hallow -- this ground. The brave men, living and dead, who struggled here,
have consecrated it, far above our poor power to add or detract. The world will
little note, nor long remember what we say here, but it can never forget what
they did here. It is for us the living, rather, to be dedicated here to the
```

```
unfinished work which they who fought here have thus far so nobly advanced. It
is rather for us to be here dedicated to the great task remaining before us --
that from these honored dead we take increased devotion to that cause for which
they gave the last full measure of devotion -- that we here highly resolve that
these dead shall not have died in vain -- that this nation, under God, shall
have a new birth of freedom -- and that government of the people, by the people,
for the people, shall not perish from the earth.'
```

```
Gettysburg_clean <- str_split(str_squish(str_replace_all(Gettysburg, "\\,|\\n|\\-|\\.", "")),
                              pattern = "\\s+")[[1]] # getting each word by itself and battle-field is
Gettysburg_length <- length(Gettysburg_clean) # getting the amount of words inside the Gettysburg addre
Gettysburg_count <- sum(str_count(Gettysburg_clean)) #getting the number of letters in the Gettysburg a
(Gettysburg_mean <-Gettysburg_count/Gettysburg_length) # dividing the amount of letters by the # of wor
```

```
## [1] 4.271375
```