Problem Set #8

Marwin Carmo

2023-11-26

Overview:

The purpose of this problem set is for you to understand how the backslash escape character (\) works in R strings, as well as to practice writing regular expressions. You will be using the `str_view_all()` function to see all the matches from your regex. You'll get practice combining character classes, quantifiers, anchors, ranges, groups, and more to build your regular expressions for each question.

Part I: Backslash (\) escape character

In this section, you will practice working with strings that include backslashes, such as for escaping characters or for writing special characters. You will be using both the `print()` and `writeLines()` functions to print out your string and compare the difference. This section is not specific to/does not involve regular expressions.

/1

1. Create a short string (could be a phrase or sentence) that contains both the single quote (`'`) and double quote (`"`) inside your string, and save it as an object called `string_with_quotes`. Use both `print()` and `writeLines()` to print out your string.

   Hint: You will need to use a backslash to escape either the single quote (`'`) or the double quote (`"`) depending on if you used single or double quotes to enclose your string.

```r
string_with_quotes <- "He then said, \"What is this 'thing'?\""
print(string_with_quotes)
```

```
[1] "He then said, \"What is this 'thing'?\""
```

```
writeLines(string_with_quotes)
```

```
He then said, "What is this 'thing'?"
```

/1

2. Create a short string (could be a phrase or sentence) that contains both the tab and newline special characters, and save it to `string_with_spchars`. Use both `print()` and `writeLines()` to print out your string.

```
string_with_spchars <- "This string contais \ttab and \nnewline."
print(string_with_spchars)
```

```
[1] "This string contais \ttab and \nnewline."
```

```
writeLines(string_with_spchars)
```

```
This string contais     tab and
newline.
```

/1

3. Create a string that contains your first name where each letter is separated by a backslash (e.g., `y\o\u\r\n\a\m\e`), and save it to `string_with_backslashes`. Use both `print()` and `writeLines()` to print out your string.

   Hint: Your `writeLines()` output should show single backslashes between each letter of your name.

```
string_with_backslashes <- "m\\a\\r\\w\\i\\n"
print(string_with_backslashes)
```

```
[1] "m\\a\\r\\w\\i\\n"
```

```
writeLines(string_with_backslashes)
```

```
m\a\r\w\i\n
```

/1

4. With respect to the previous questions, explain in general why the output created by the `print()` function differs from the output created by the `writeLines()` function.

The `print()` function displays the content of an object, while `writeLines()` write character strings.

Part II: Matching characters

In this section and the next, you will practice writing regular expressions to match specific text. Use `str_view_all()` for all the following questions to show the matches.

/1

1. Show all matches to single quotes (`'`) in your `string_with_quotes` that you created in Part I.

```
stringr::str_view_all(string_with_quotes, "\'")
```

```
Warning: `str_view()` was deprecated in stringr 1.5.0.
i Please use `str_view_all()` instead.
```

```
[1] | He then said, "What is this <'>thing<'>?"
```

/1

2. Show all matches to double quotes (`"`) in `string_with_quotes`.

```
stringr::str_view_all(string_with_quotes, "\"")
```

```
[1] | He then said, <">What is this 'thing'?<">
```

/1

3. Show all matches to tab characters in `string_with_spchars`.

```
stringr::str_view_all(string_with_spchars, "\t")
```

```
[1] | This string contais <{\t}>tab and
    | newline.
```

/1

4. Show all matches to newline characters in `string_with_spchars`.

```r
stringr::str_view_all(string_with_spchars, "\n")
```

```
[1] │ This string contais {\t}tab and <
    │ >newline.
```

/1

5. Show all matches to backslashes (\) in `string_with_backslashes`.

```r
stringr::str_view_all(string_with_backslashes, "\\\\")
```

```
[1] │ m<\>a<\>r<\>w<\>i<\>n
```

Part III: Regular expressions

/1

1. Copy the following code to create the character vector `text`:

   ::: {.cell}

   ```r
   text <- c("In 5... 4... 3... 2...",
             "It can cost anywhere between $50 to $100 (... or even $1k!)",
             "These are parenthesis (), while these are brackets []... I think.")
   ```

   :::

/1

2. Show all matches to a capital `I` at the beginning of the string.

```r
stringr::str_view_all(text, "^I")
```

```
[1] │ <I>n 5... 4... 3... 2...
[2] │ <I>t can cost anywhere between $50 to $100 (... or even $1k!)
[3] │ These are parenthesis (), while these are brackets []... I think.
```

/1

3. Show all matches to a period at the end of the string.

```
stringr::str_view_all(text, "\\.$")
```

```
[1] | In 5... 4... 3... 2..<.>
[2] | It can cost anywhere between $50 to $100 (... or even $1k!)
[3] | These are parenthesis (), while these are brackets []... I think<.>
```

/1

4. Show all matches to 1 or more digits.

```
stringr::str_view_all(text, "\\d{1,}")
```

```
[1] | In <5>... <4>... <3>... <2>...
[2] | It can cost anywhere between $<50> to $<100> (... or even $<1>k!)
[3] | These are parenthesis (), while these are brackets []... I think.
```

/2

5. Show all matches to all dollar amounts, including the dollar sign and `k` if there is one (i.e., `$50`, `$100`, `$1k`)

```
stringr::str_view_all(text, "\\$(\\d|k)+")
```

```
[1] | In 5... 4... 3... 2...
[2] | It can cost anywhere between <$50> to <$100> (... or even <$1k>!)
[3] | These are parenthesis (), while these are brackets []... I think.
```

/1

6. Show all matches to ellipses (. . .)

```
stringr::str_view_all(text, "\\...")
```

```
[1] | In 5<...> 4<...> 3<...> 2<...>
[2] | It can cost anywhere between $50 to $100 (<...> or even $1k!)
[3] | These are parenthesis (), while these are brackets []<...> I think.
```

/2

7. Show all matches to parentheses, including the contents between the parentheses if there are any.

```
stringr::str_view_all(text, "\\(.*\\)")
```

```
[1] | In 5... 4... 3... 2...
[2] | It can cost anywhere between $50 to $100 <(... or even $1k!)>
[3] | These are parenthesis <()>, while these are brackets []... I think.
```

/2

8. Show all matches to words (define words as containing only letters, upper or lowercase)

```
stringr::str_view_all(text, "\\b[aA-zZ]+\\b")
```

```
[1] | <In> 5... 4... 3... 2...
[2] | <It> <can> <cost> <anywhere> <between> $50 <to> $100 (... <or> <even> $1k!)
[3] | <These> <are> <parenthesis> (), <while> <these> <are> <brackets> []... <I> <think>.
```

/2

9. Show all matches to either a word that's 4 or more letters long *or* ellipses.

```
stringr::str_view_all(text, "\\b[aA-zZ]{4,}\\b|\\...")
```

```
[1] | In 5<...> 4<...> 3<...> 2<...>
[2] | It can <cost> <anywhere> <between> $50 to $100 (<...> or <even> $1k!)
[3] | <These> are <parenthesis> (), <while> <these> are <brackets> []<...> I <think>.
```

/3

10. Show all matches to any digit or vowel (upper or lowercase) that repeats 2 times in a row (i.e., the same digit or vowel repeated twice in a row)

```
stringr::str_view_all(text, "(\\d|[aeiouAEIOU])\\1")
```

```
[1] | In 5... 4... 3... 2...
[2] | It can cost anywhere betw<ee>n $50 to $1<00> (... or even $1k!)
[3] | These are parenthesis (), while these are brackets []... I think.
```