````
---
title: "Week4_Homework"
author: "Angus Huang"
date: "September 20, 2015"
output: html_document
---
4. Describe the types of strings that conform to the following regular expressions and
construct an example that is matched by the regular expression.
(a) [0-9]+\\$
(b) \\b[a-z]{1,4}\\b
(c) .*?\\.txt$
(d) \\d{2}/\\d{2}/\\d{4}
(e) <(.+?)>.+?</\\1>

Answer:
a)
[0-9] denotes the type of strings with numerical values only.
+ denotes matching the preceeding item one or more time.
"$" means to find item with symbol $ while suppressing metacharacter $'s meaning
For examples:
```{r}
library (stringr)
test <- c(" Good banana sells from $200 through 249$ when lucky for10$ $ here is indian rupee$")
anw <-unlist (str_extract_all(test, "[0-9]+\\$"))
anw
```

b)
[a-z] means alphabetic values from a to z
{1,4} means to find the preceeding item one time but not more than 4 times.
\\b means word edge therefore, the whole parameters will only exatract words (alphabetic ) with 4 letters

```{r}
anw <-unlist (str_extract_all(test, "\\b[a-z]{1,4}\\b"))
anw

```
(c) .*?\\.txt$
. matches any single character
*

```{r}
test1 <- c(" Good banana sells from $200 through 249$. When lucky for10$ $ here is indian rupee$ filename1.txt, randomtxt.txt")
anw <-unlist (str_extract_all(test1, ".*?\\.txt$"))
anw
```
d) \\d{2}/\\d{2}/\\d{4}
Answer:
\d means digit values from 9-0 and {2} denotes the number of length
/.../ contains a regex
{2} and {4} denotes the number of digits in consecutive matches
The whole
```{r}
test2 <- c(" Good banana sells from $20000 through 249$. When lucky for10$ $ here is indian rupee$ filename1.txt, randomtxt.txt call 20 19 1817 on
01/01/2014 is your lucky day")
anw <-unlist (str_extract_all(test2, "\\d{2}/\\d{2}/\\d{4}"))
anw
```



5. Rewrite the expression [0-9]+\\$ in a way that all elements are altered but the
expression performs the same task.
```{r}
anw <- unlist (str_extract_all(test2,"(\\d){1,}[$]"))
anw
```
6. Consider the mail address chunkylover53[at]aol[dot]com.
(a) Transform the string to a standard mail format using regular expressions.
(b) Imagine we are trying to extract the digits in the mail address. To do so we write
the expression [:digit:]. Explain why this fails and correct the expression.
(c) Instead of using the predefined character classes, we would like to use the predefined
symbols to extract the digits in the mail address. To do so we write the expression
  \\D Explain why this fails and correct the expression.
```{r}
email <- c("chunkylover53[at]aol[dot]com")
temail <- gsub("\\[at]","@",email )
temail
temail <-gsub("\\[dot]",".",temail)
temail
```
b)
Answer: [:digit:] operator only catpaures a single digit or treat each digit as a single item. It is missing the length identifier.
```{r}
str_extract_all (email,"[:digit:]{1,}")
str_extract (email, "\\d{1,}")
```
c)
Answer: \\D capital D here denotes no digit and thus will extract all alphabet values. And length identifier needs to be added.

```{r}
str_extract_all (email,"\\d+")
```



---
title: "Week4Homework_Angus"
author: "Angus Huang"
date: "September 20, 2015"
output: html_document
---

4. Describe the types of strings that conform to the following regular expressions and
construct an example that is matched by the regular expression.
(a) [0-9]+\\$
(b) \\b[a-z]{1,4}\\b
````

(c) .*?\\.txt$
(d) \\d{2}/\\d{2}/\\d{4}
(e) <(.+?)>.+?</\\1>

4.a)
Answer:
a)
[0-9] denotes the type of strings with numerical values only.
+ denotes matching the preceeding item one or more time.
"\\$" means to find item with symbol $ while suppressing metacharacter $'s meaning.

For examples:
```{r}
library (stringr)
test <- c(" Good banana sells from $200 to 249$ when lucky for10$ $ here means a fictious dollars$")
anw <-unlist (str_extract_all(test, "[0-9]+\\$"))
anw
```

b)
[a-z] means alphabetic values from a to z

{1,4} means to find the preceeding item one time but not more than 4 times.

\\b means word edge the whole parameters will only exatract whole words

```{r}
anw <-unlist (str_extract_all(test, "\\b[a-z]{1,4}\\b"))
anw

```

(c) .*?\\.txt$

. matches any single character, * means preceding item will be matched zero or more times, ? means preceding item is optional with matching at most once
$ denotes teh ends of sentences with word txt


```{r}
test1 <- c("badfilename txt", "propername.txt")

anw <-unlist (str_extract_all(test1, ".*?\\.txt$"))
anw
```

d) \\d{2}/\\d{2}/\\d{4}

Answer:
\d means digit values from 9-0 and {2} denotes the number of length
/.../ contains a regex

{2} and {4} denotes the number of digits in consecutive matches, making the code to extract date value with slash format.

```{r}
test2 <- c(" Good banana sells from $20000 through 249$. When lucky for10$ $ here is indian rupee$ filename1.txt, randomtxt.txt call 20 19 1817 on 01/01/2014 is your lucky day")
anw <-unlist (str_extract_all(test2, "\\d{2}/\\d{2}/\\d{4}"))
anw
```


(e)

() is for grouping
.+? means everything and anything but only once

< beginning of word
greater than sign means end of word

\\1 is for backreferencing

Please note : the code in this question is making rmarkdown not populating certian part of this file. A pdf version is included as attachement.
 Greater or less than sign is causing anything prior to be hidden.
```{r}
test3 <- c("<abc> This is a blog on banana</abc> ")

anw <-unlist (str_extract_all(test3, "<(.+?)>.+?</\\1>"))
anw
```


5. Rewrite the expression [0-9]+\\$ in a way that all elements are altered but the
expression performs the same task.

```{r}
test2 <- c(" Good banana sells from $20000 through 249$. When lucky for10$ $ here is indian rupee$ filename1.txt, randomtxt.txt call 20 19 1817 on 01/01/2014 is your lucky day")
anw <- unlist (str_extract_all(test2,"(\\d){1,}[$]"))
anw

```

6. Consider the mail address chunkylover53[at]aol[dot]com.
(a) Transform the string to a standard mail format using regular expressions.
(b) Imagine we are trying to extract the digits in the mail address. To do so we write
the expression [:digit:]. Explain why this fails and correct the expression.
(c) Instead of using the predefined character classes, we would like to use the predefined
symbols to extract the digits in the mail address. To do so we write the expression
  \\D Explain why this fails and correct the expression.

a) answer:
```{r}
email <- c("chunkylover53[at]aol[dot]com")
temail <- gsub("\\[at]","@",email )
temail
```

```
temail <-gsub("\\[dot]",".",temail)
temail
```
b)
Answer: [:digit:] operator only catpaures a single digit or treat each digit as a single item. It is missing the length identifier.
```{r}
str_extract_all (email,"[:digit:]{1,}")
str_extract (email, "\\d{1,}")
```
c)
Answer: \\D capital D here denotes no digit and thus will extract all alphabet values. And length identifier needs to be added.

```{r}
str_extract_all (email,"\\d+")
```