Georgia State University | UNIVERSITY LIBRARY

RESEARCH DATA SERVICES
LIBRARY.GSU.EDU/DATA

Regular Expressions

A language for defining patterns in digital texts

https://research.library.gsu.edu/dataservices

Jeremy Walker
Data Services Librarian
jwalker184@gsu.edu

Research Data Services

RDS Team Members:

- Dr. Swygart-Hobaugh: Qualitative Methods
- Joel Glogowski: Data Visualization
- Kelsey Jordan: Data Visualization
- Jeremy Walker: Quantitative Methods

General Information, workshops, contact info:

https://research.library.gsu.edu/dataservices

Data Analysis Tools & Methods

TOOLS for DATA ANALYSIS

Quantitative/Statistical

- Excel JG RA JW KJ
- SPSS MS RA GU
- SAS RA
- Stata RA GU
- R RA JW
- Python JW
- Mplus RA
- Stat/Transfer RA
- RDSAT RA

Qualitative

- NVivo (qualitative data analysis tool) MS

METHODS for DATA ANALYSIS

- Basic Statistical Analysis MS RA GU JW
- Structural Equation Modeling (SEM) RA
- Hierarchical Linear Modeling (HLM) RA
- Network Analysis RA GU
- Cross-Sectional and Longitudinal Data RA
- Mixed Methods RA
- Text Mining/Natural Language Processing (NLP) JW
- Clustering/Segmentation JW
- Webscraping JW
- Qualitative Data Analysis MS

NOTE: Please see the RDS Scope of Services statement.

Mapping & Data Visualization

MAPPING & DATA VISUALIZATION

- Social Explorer & SimplyAnalytics JG
- PolicyMap JG
- Infographics RA
- Tableau data visualization JG KJ JW MS
- GIS resources*
- For access to ESRI's Business Analyst Online and Community Analyst databases, please contact John Tougas in the Geosciences Department.

* The GSU Library currently does not have any one that can help with ArcGIS or other GIS mapping tools. If you want future updates on whether the Library will be adding GIS support to its services, please contact Mandy Swygart-Hobaugh, the Leader of the Research Data Services Team.

Finding Data & Statistics

FINDING DATA & STATISTICS

- Social science data (ICPSR, qualitative data, & more) MS RA
- Business data, consumer and marketing data, and international trade data JG JW
- Bloomberg Terminal for real-time financial market data JG

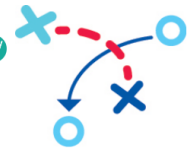
Survey Design

SURVEY DESIGN

- Qualtrics survey tool -- mechanics (not methods) of using Qualtrics to create surveys MS RA JW KJ
- Survey Design and implementation assistance (survey/questionnaire design; survey implementation; survey analysis; applied sampling; total survey error) RA
- Finding existing surveys and measurement scales for data collection MS RA

Data Management

DATA MANAGEMENT



- Data cleaning RA JW
- Database setup RA JW
- Support for writing data management plans for research grant proposals using the DMPTool, and for managing research data throughout the research data life cycle MS RA

Writing a Grant and Need Boilerplate Language about Library Resources?

- Grant Proposals - Boilerplate Language about GSU Library Support and Resources

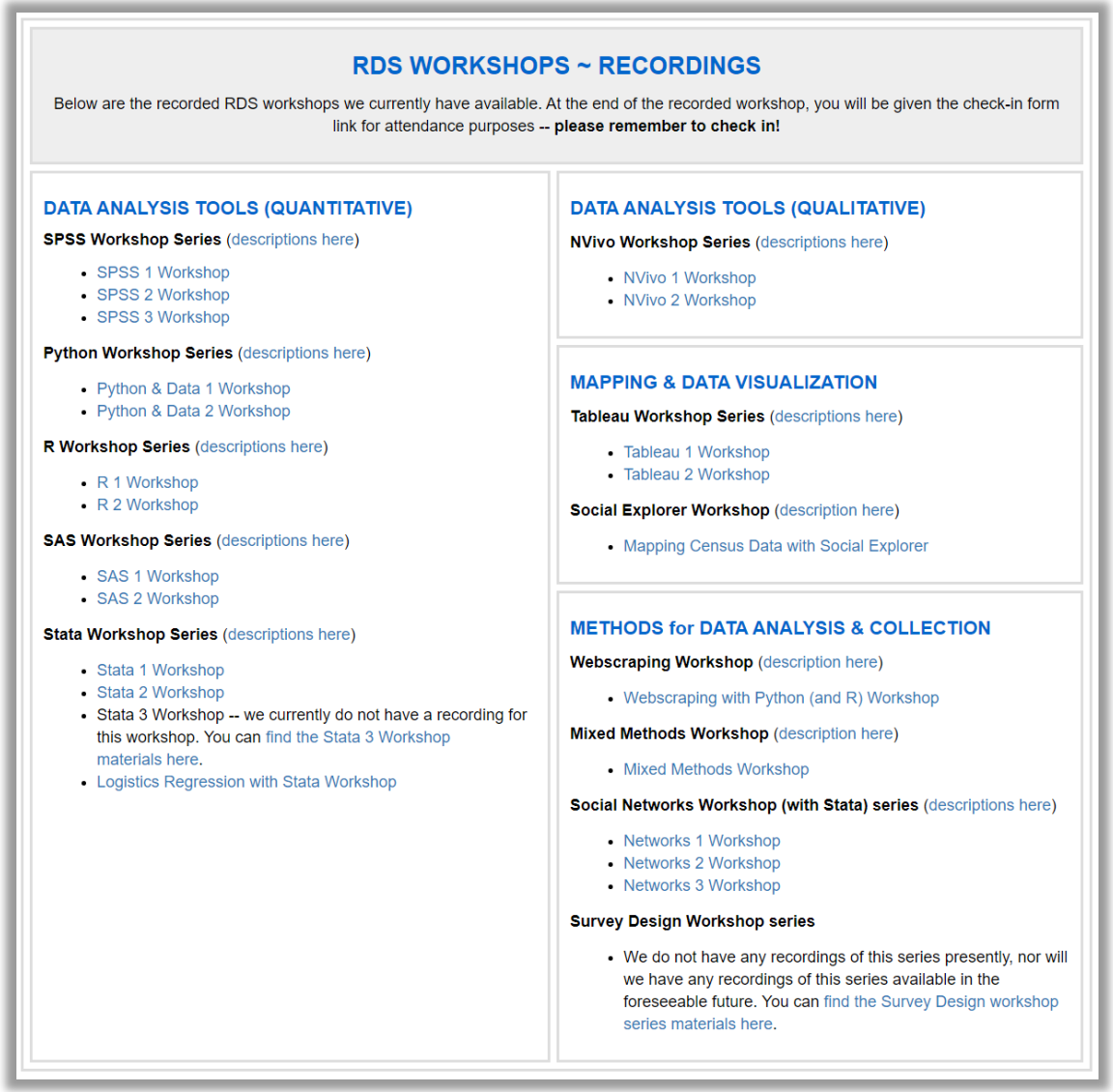
Other Recorded Workshops

The Research Data Services team hosts a variety of workshops for different software tools and packages.

At the moment, we are also uploading recorded versions of our workshops to our website for anyone to access.

Included in this is a generalized R workshop series that covers much of the same content as today's session.

https://research.library.gsu.edu/dataservices/rds-workshops-recordings



RDS WORKSHOPS ~ RECORDINGS

Below are the recorded RDS workshops we currently have available. At the end of the recorded workshop, you will be given the check-in form link for attendance purposes -- **please remember to check in!**

DATA ANALYSIS TOOLS (QUANTITATIVE)

SPSS Workshop Series (descriptions here)

- SPSS 1 Workshop
- SPSS 2 Workshop
- SPSS 3 Workshop

Python Workshop Series (descriptions here)

- Python & Data 1 Workshop
- Python & Data 2 Workshop

R Workshop Series (descriptions here)

- R 1 Workshop
- R 2 Workshop

SAS Workshop Series (descriptions here)

- SAS 1 Workshop
- SAS 2 Workshop

Stata Workshop Series (descriptions here)

- Stata 1 Workshop
- Stata 2 Workshop
- Stata 3 Workshop -- we currently do not have a recording for this workshop. You can find the Stata 3 Workshop materials here.
- Logistics Regression with Stata Workshop

DATA ANALYSIS TOOLS (QUALITATIVE)

NVivo Workshop Series (descriptions here)

- NVivo 1 Workshop
- NVivo 2 Workshop

MAPPING & DATA VISUALIZATION

Tableau Workshop Series (descriptions here)

- Tableau 1 Workshop
- Tableau 2 Workshop

Social Explorer Workshop (description here)

- Mapping Census Data with Social Explorer

METHODS for DATA ANALYSIS & COLLECTION

Webscraping Workshop (description here)

- Webscraping with Python (and R) Workshop

Mixed Methods Workshop (description here)

- Mixed Methods Workshop

Social Networks Workshop (with Stata) series (descriptions here)

- Networks 1 Workshop
- Networks 2 Workshop
- Networks 3 Workshop

Survey Design Workshop series

- We do not have any recordings of this series presently, nor will we have any recordings of this series available in the foreseeable future. You can find the Survey Design workshop series materials here.

Data Certification

(shameless self promotion)

- Attend 5 different workshops this semester
- Earn a data certification through Research Data Services

https://research.library.gsu.edu/dataservices/data-certified

Today's Agenda

- Introduction to Regular Expressions:
  - Concepts
  - Use Cases

- Guided walkthrough using R
  - Using the Tidyverse package s*tringr*

  - Pattern matching using…
    - Basic and fixed strings
    - Special and reserved characters
    - Repetitions and quantifiers
    - Alternates and groups
    - Positional methods (anchors and lookarounds)

  - Practice on both placeholder texts and real texts

Pre-requisites:

This module / video assumes that you have working knowledge of R and RStudio.

If you need a refresher, please refer to the earlier slide containing links to recorded R workshops.

Regular Expressions

HIGH LEVEL NOTES

- Sometimes referred to as “RegEx” or “RegExp”.
- **Used for programmatically detecting patterns in text.**
- Different formats and formulations exist. But once you learn one, it is easy to learn any other system.
  - Pepsi vs. Coke
  - Netflix vs. Hulu
  - Python vs. R
  - SPSS vs. Stata
- This stuff can be *weird* and *hard* to wrap your mind around at first. Stick with it!

Regular Expressions

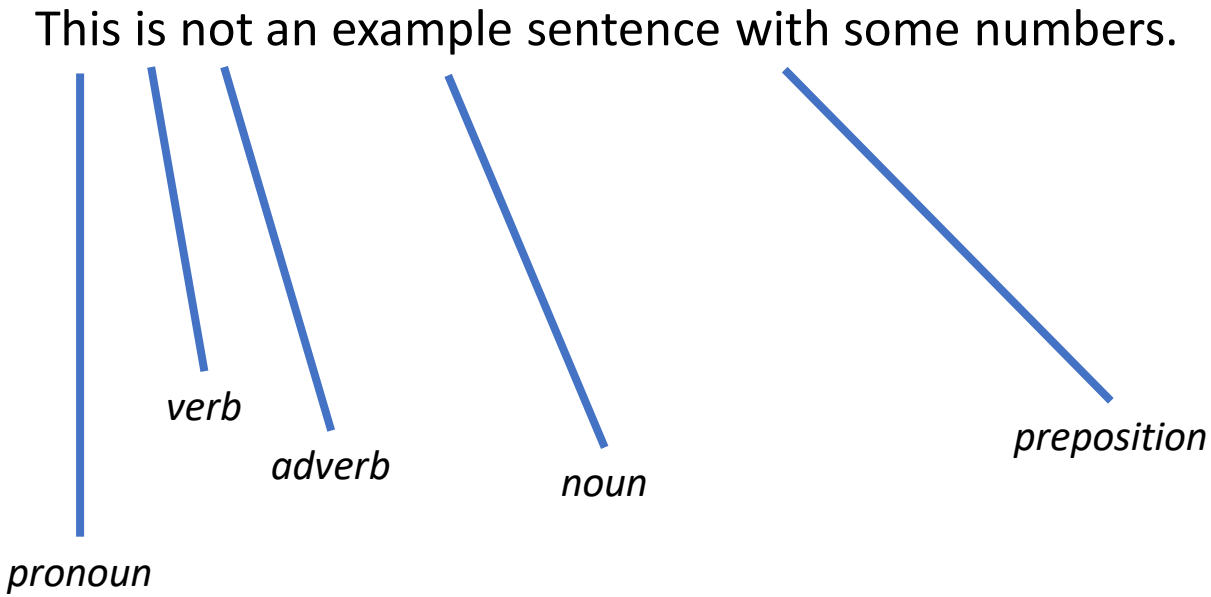
Patterns?

Beyond simple keyword searches, regular expressions can match complex patterns in text:

- Abstract forms of words and phrases
- Context dependent patterns

Regular Expressions

Patterns?

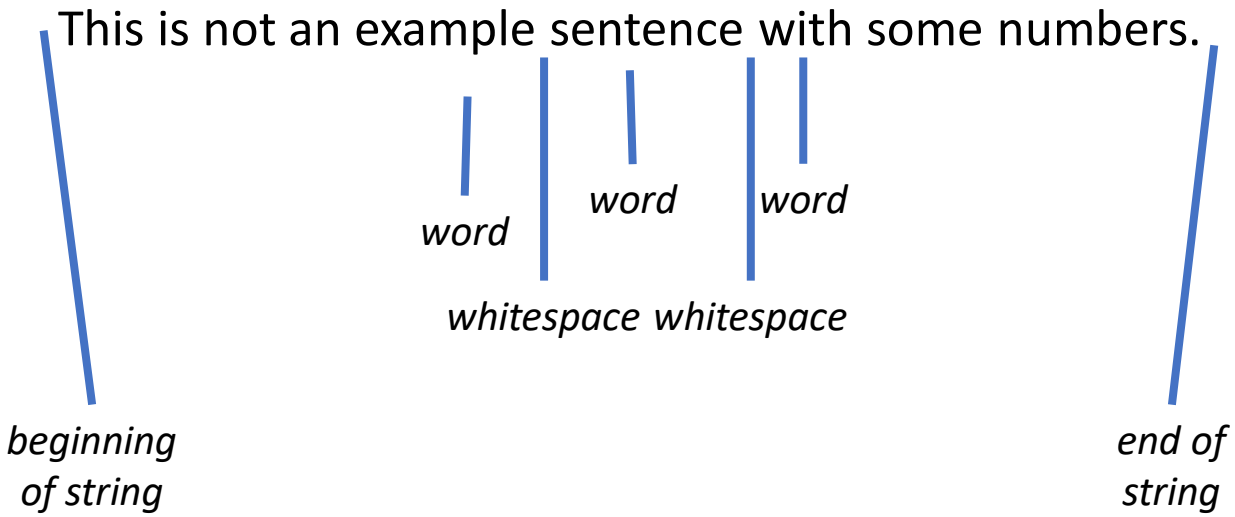


Beyond simple keyword searches, regular expressions can match complex patterns in text:

- Abstract forms of words and phrases
- Context dependent patterns

Regular Expressions

Patterns?



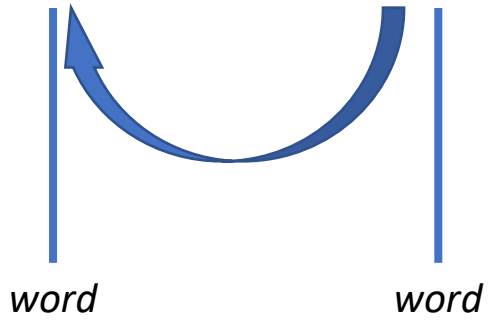
Beyond simple keyword searches, regular expressions can match complex patterns in text:

- Abstract forms of words and phrases
- Context dependent patterns

Regular Expressions

Patterns?

This is not an example sentence with some numbers.



Beyond simple keyword searches, regular expressions can match complex patterns in text:

- Abstract forms of words and phrases
- Context dependent patterns

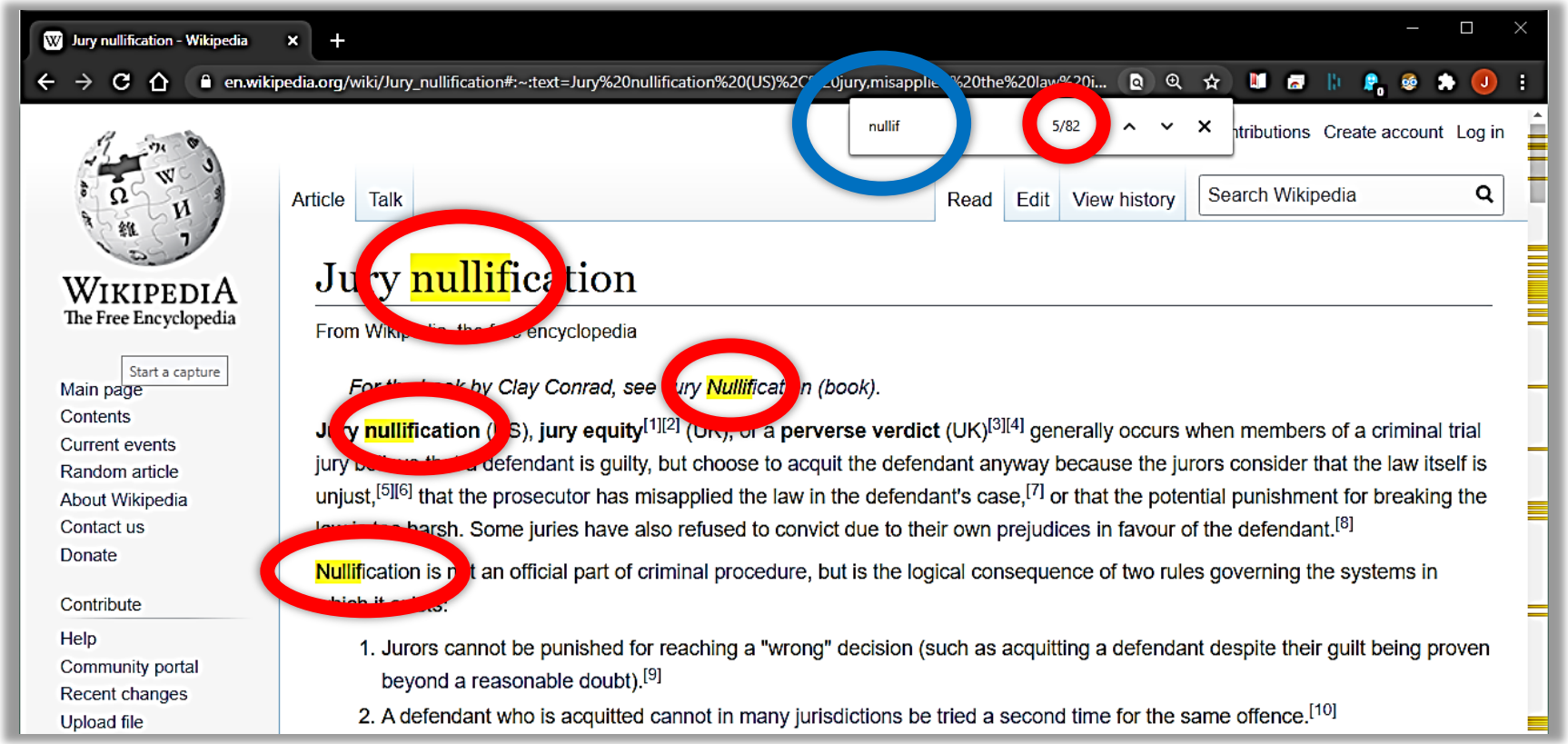
Regular Expressions

Use Cases

- Identifying texts that contain specific patterns
- Extracting contents from texts
- Using regular expression matches / outputs to define structured data for analysis

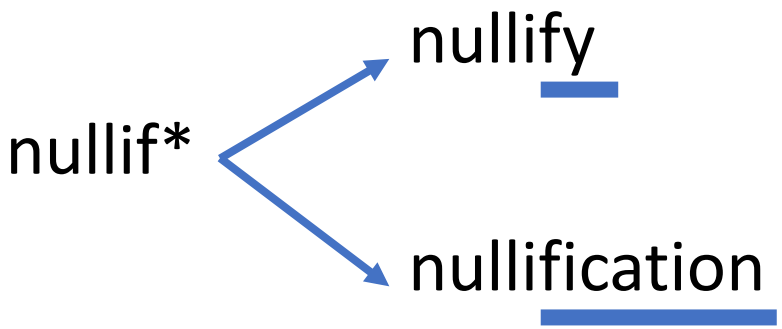
Familiar Examples

Using the “Find” feature on webpages and documents.



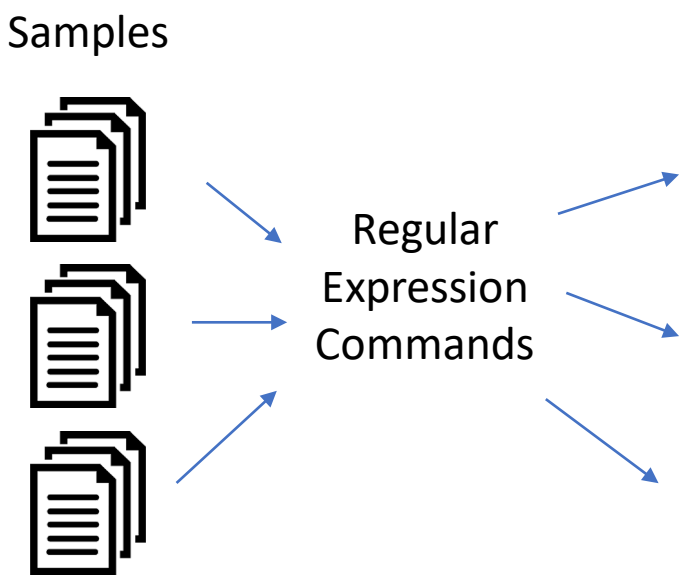
Familiar Examples

Using wildcards (*) in research databases



Regular Expressions

Use Cases



Data Table

Variable1	Variable2	Variable3
“extracted text”	TRUE	1
NULL	FALSE	1
“extracted text”	FALSE	0

- Identifying texts that contain specific patterns
- Extracting contents from texts
- Using regular expression matches / outputs to define structured data for analysis

Regular Expressions

Stringr

Stringr is part of the Tidyverse suite of R packages. Although not strictly necessary for using regular expressions, this package provides a lot of pre-built functions and utilities that are extremely useful.

- Stringr Homepage - https://stringr.tidyverse.org/

- Stringr Cheat sheet:
  - GitHub Page
  - Direct PDF Download

Example Text: “DEATH PENALTY”

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

D	E	A	T	H		P	E	N	A	L	T	Y
DEATH					.	.	.	.	.	.	.	.
DEATH					\s	\w	\w	\w	\w	\w	\w	\w

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: “DEATH PENALTY”

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

D	E	A	T	H		P	E	N	A	L	T	Y
DEATH					.	.	.	.	.	.	.	.
DEATH					\s	\w	\w	\w	\w	\w	\w	\w
DEATH					\s	\w{7}						
DEATH					\s	\w{1,7}						

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: “DEATH PENALTY”

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

D	E	A	T	H		P	E	N	A	L	T	Y
DEATH					.	.	.	.	.	.	.	.
DEATH					\s	\w	\w	\w	\w	\w	\w	\w
DEATH					\s	\w{7}						
DEATH					\s	\w{1,7}						
DEATH					\s	\w+						
DEATH					\s	\w*						

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: “DEATH PENALTY”

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

D	E	A	T	H		P	E	N	A	L	T	Y
DEATH					.	.	.	.	.	.	.	.
DEATH					\s	\w	\w	\w	\w	\w	\w	\w
DEATH					\s	\w{7}						
DEATH					\s	\w{1,7}						
DEATH					\s	\w+						
DEATH					\s	\w*						
DEATH					.{1,50}							

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: Dates

What is one pattern that could describe all of the examples below?

4/12/1999

12/26/2015

5/5/87

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: Dates

What is one pattern that could describe all of the examples below?

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

Regular expressions are like any other challenge. The more you can isolate and separate different elements, the easier it will be to find a solution.

4 / 12 / 1999

12 / 26 / 2015

5 / 5 / 87

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: Dates

What is one pattern that could describe all of the examples below?

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

Regular expressions are like any other challenge. The more you can isolate and separate different elements, the easier it will be to find a solution.

4 / 12 / 1999

12 / 26 / 2015

5 / 5 / 87

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: Dates

What is one pattern that could describe all of the examples below?

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

4	/	12	/	1999
12	/	26	/	2015
5	/	5	/	87

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: Dates

What is one pattern that could describe all of the examples below?

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (**\S** *for non-whitespaces*)
\d	any digit (**\D** *for non-digits*)
\w	any word character (**\W** *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

4	/	12	/	1999
12	/	26	/	2015
5	/	5	/	87
\d{1,2}	/	\d{1,2}	/	\d{2,4}

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: Dates

What is one pattern that could describe all of the examples below?

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (*\S for non-whitespaces*)
\d	any digit (*\D for non-digits*)
\w	any word character (*\W for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

4	/	12	/	1999
12	/	26	/	2015
5	/	5	/	87
\d{1,2}	/	\d{1,2}	/	\d{2,4}
\d+	/	\d+	/	\d+

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n, }	**n** or more
a{n, m}	between **n** and **m**

Example Text: Dates

What is one pattern that could describe all of the examples below?

regexp (to mean this)	matches (which matches this)
a (etc.)	a (etc.)
\.	.
\!	!
\?	?
\\	\
\(	(
\)	)
\{	{
\}	}
\n	new line (return)
\t	tab
\s	any whitespace (\S *for non-whitespaces*)
\d	any digit (\D *for non-digits*)
\w	any word character (\W *for non-word chars*)
\b	word boundaries
[:digit:] [1]	digits
[:alpha:] [1]	letters
[:lower:] [1]	lowercase letters
[:upper:] [1]	uppercase letters
[:alnum:] [1]	letters and numbers
[:punct:] [1]	punctuation
[:graph:] [1]	letters, numbers, and punctuation
[:space:] [1]	space characters (i.e. \s)
[:blank:] [1]	space and tab (but not new line)
.	every character except a new line

4	/	12	/	1999
12	/	26	/	2015
5	/	5	/	87
\d{1,2}	/	\d{1,2}	/	\d{2,4}
\d+	/	\d+	/	\d+

Standard RegEx: “\d+/\d+/\d+”

R / Stringr: “\\d+/\\d+/\\d+”

regexp	matches
a?	zero or one
a*	zero or more
a+	one or more
a{n}	exactly **n**
a{n,}	**n** or more
a{n, m}	between **n** and **m**

More Resources

- R-Bloggers – Demystifying Regular Expressions in R:
  https://www.r-bloggers.com/demystifying-regular-expressions-in-r/
- R for Data Science – Chapter 14 Strings:
  https://r4ds.had.co.nz/strings.html
- StringR Offical Page:
  https://stringr.tidyverse.org/
- Introduction to StringR:
  https://cran.rstudio.com/web/packages/stringr/vignettes/stringr.html
- Regular Expressions:
  https://stringr.tidyverse.org/articles/regular-expressions.html

That’s all folks!

Contact me or visit the RDS website for questions or support

https://research.library.gsu.edu/dataservices

(for after today)

Jeremy Walker
Data Services Librarian
jwalker184@gsu.edu