Scripting Languages

Module 5

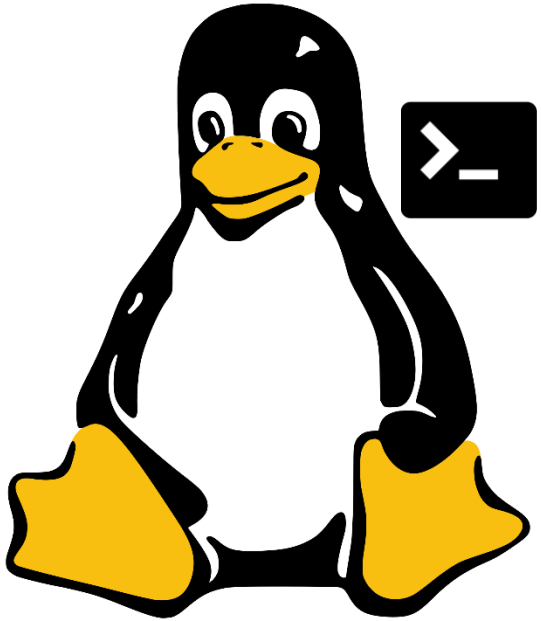
Regular Expressions,
Redirection, Piping and Grep

Contents

Learning Objectives

After completing this module, you should be able to work with:

- Regular Expressions
- Regular Expression Engines
- Grep and Regex
- Anchors and Wildcards
- Extended RegEx Engine
- ERE Repetition and Optionality
- OR and Expression Grouping
- Common Grep Options
- Piping and Redirection



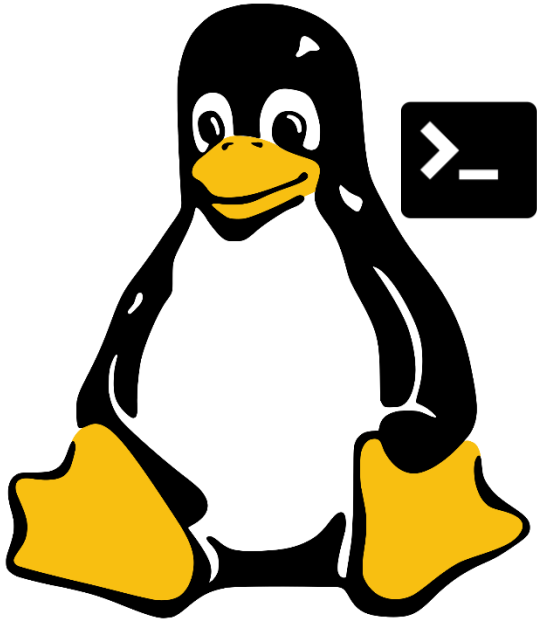
Regular Expressions

Regular Expressions

- Regular Expressions, more commonly referred to as **regex**, are used to match patterns in text
- A regex text pattern is provided to a *regex engine* to allow it to find a match
- Once a match is found, other commands and utilities can then be called upon to interact with the matched data in some way

Regex engines

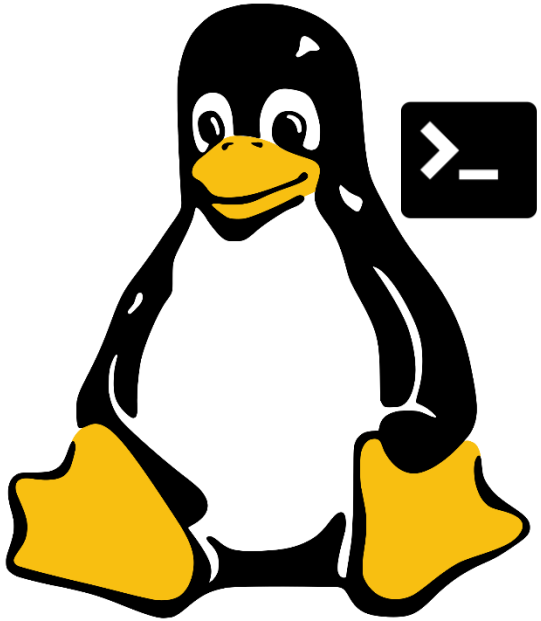
- There are two regular expression engines supported by bash commands
  - Basic Regular Expression Engine (BRE)
  - Extended Regular Expression Engine (ERE)
- These are supported by several commands and utilities, in particular **grep**, **sed** and **awk**
- **BRE** and **ERE** are also regularly used for data validation purposes as well
- The remainder of this module will examine *regex* as used with **grep**



grep

What is grep?

- **grep** stands for **G**lobal **R**egular **E**xpression **P**rint
- grep searches through stipulated input files for lines containing a match to a regex pattern
- When a match is found in a line, grep copies the line to standard output by default, or another output if stipulated by options or piping
- grep is highly integrated with BRE and ERE to perform the tasks it is designed to do



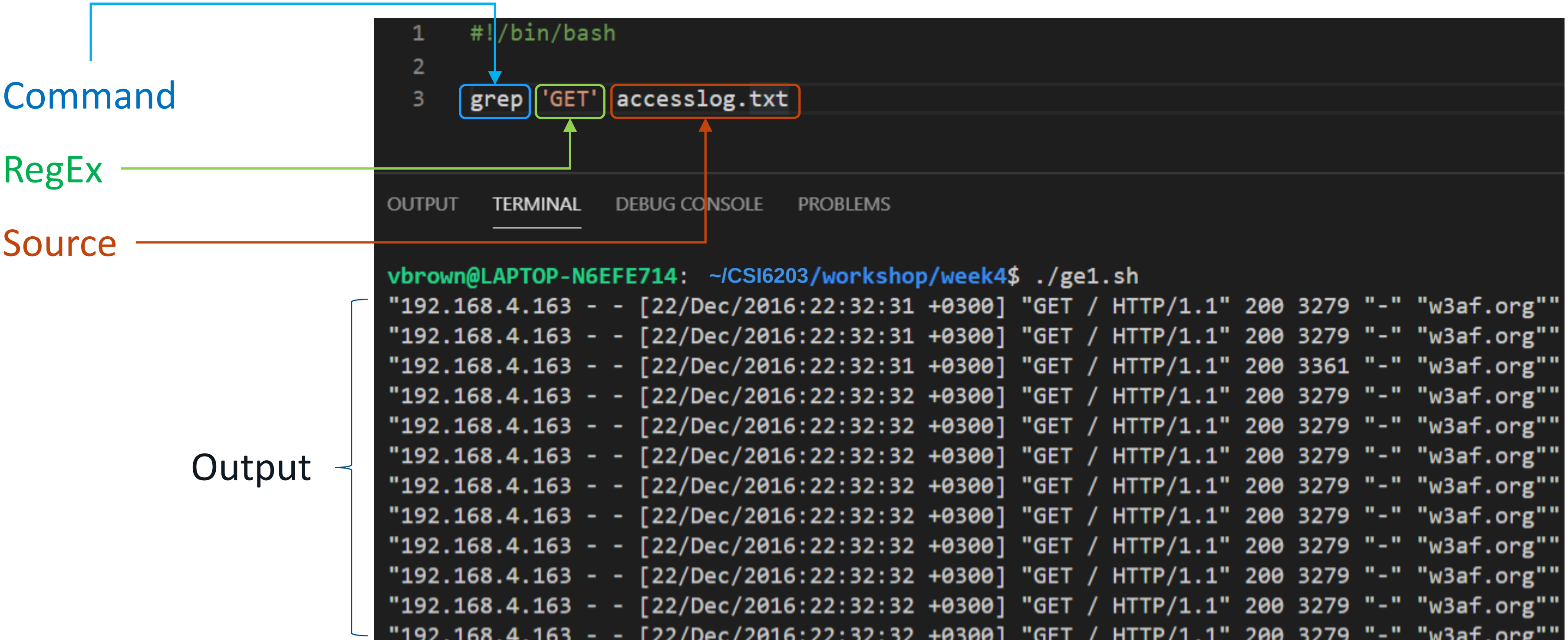
grep and **regex**

Sample Data Set

- The following grep and regex examples are based on an access log data set acquired from https://github.com/ocatak/apache-http-logs/blob/master/w3af.txt (06/07/2020)

```
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /DVWA/dvwa/css/login.css HTTP/1.1" 200 668 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /DVWA/dvwa/images/login_logo.png HTTP/1.1" 200 13161 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /images/joomla_black.gif HTTP/1.1" 200 4030 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /index.php/component/users/?view=reset HTTP/1.1" 200 3023 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /index.php?format=feed&type=rss HTTP/1.1" 200 1083 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /templates/beez_20/css/epsrnola.css HTTP/1.1" 404 525 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /templates/beez_20/css/general.css HTTP/1.1" 200 1441 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /templates/beez_20/css/opisitno.css HTTP/1.1" 404 525 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /templates/beez_20/images/minus.png HTTP/1.1" 200 452 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /templates/beez_20/images/plus.png HTTP/1.1" 200 454 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "GET /templates/beez_20/javascript/md_stylechanger.js HTTP/1.1" 200 1111 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "POST /DVWA/login.php HTTP/1.1" 302 384 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:42 +0300] "POST /index.php HTTP/1.1" 500 1924 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:43 +0300] "GET /DVWA/dvwa/ HTTP/1.1" 200 730 "http://192.168.4.161/" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:35:43 +0300] "GET /DVWA/dvwa/css/ HTTP/1.1" 200 755 "http://192.168.4.161/" "w3af.org""
```

Simple BRE matching



Command

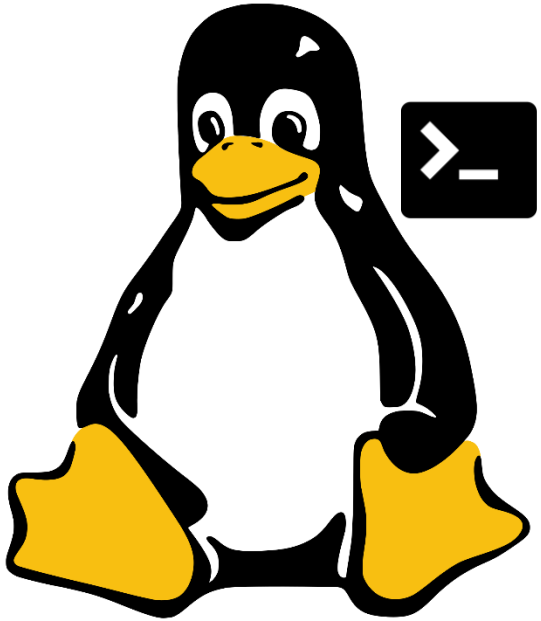
RegEx

Source

Output

```bash
#!/bin/bash

grep 'GET' accesslog.txt
```

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge1.sh
"192.168.4.163 - - [22/Dec/2016:22:32:31 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:31 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:31 +0300] "GET / HTTP/1.1" 200 3361 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
"192.168.4.163 - - [22/Dec/2016:22:32:32 +0300] "GET / HTTP/1.1" 200 3279 "-" "w3af.org""
```



Anchors and Wildcards

Anchor characters

- Regex patterns can use special characters called anchor points to represent specific locations within the text
- The two most common anchor points are
  - The start of the line ‘**^**’
    (The **^** symbol is the circum accent, or circum for short)
  - The end of the line ‘**$**’

Start of line anchor ^

start of line (circum) anchor

```
#!/bin/bash

grep '^/DVWA' accesslogdir.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge2.sh
/DVWA HTTP/1.1" 301 573 "-" "w3af.org""
/DVWA/ HTTP/1.1" 302 469 "http://192.168.4.161/" "w3af.org""
/DVWA/ HTTP/1.1" 302 384 "http://192.168.4.161/" "w3af.org""
/DVWA/ HTTP/1.1" 302 384 "http://192.168.4.161/" "w3af.org""
/DVWA/dvwa/css/login.css HTTP/1.1" 200 668 "http://192.168.4.161/" "w3af.org""
/DVWA/dvwa/css/olign.css HTTP/1.1" 404 514 "-" "w3af.org""
/DVWA/dvwa/images/login_logo.png HTTP/1.1" 200 13161 "http://192.168.4.161/" "w3af.org""
/DVWA/dvwa/images/olign_olog.png HTTP/1.1" 404 522 "-" "w3af.org""
/DVWA/login.php HTTP/1.1" 200 986 "http://192.168.4.161/" "w3af.org""
/DVWA/login.php HTTP/1.1" 200 986 "http://192.168.4.161/" "w3af.org""
/DVWA/olign.php HTTP/1.1" 404 505 "-" "w3af.org""
```

Output

End of line anchor $

end of line ($) anchor

```
#!/bin/bash

grep '503$' accesslogdir.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge3.sh
/4MlfjsG9.py HTTP/1.1" 404 503
/cdcZMNEA.cgi HTTP/1.1" 404 503
/oDZI69nQ.do HTTP/1.1" 404 503
/F2UuPWfX.pl HTTP/1.1" 404 503
/toAvZOBg.rb HTTP/1.1" 404 503
/iAio8STI.py HTTP/1.1" 404 503
/O1LD47Jq.rb HTTP/1.1" 404 503
/BA5sNSzq.do HTTP/1.1" 404 503
```

Output

Wildcard characters

- Wildcards are characters that could match a range of characters
- In regex, the most common wildcard is dot '**.**'
- The *dot* character can be used to represent any character, e.g. find lines that start with a string ending in *an*

```
1   #!/bin/bash
2
3   grep -i '^.an' wild1.txt
```

1 2 3 4

1. Make match case insensitive
2. Must occur at start of line
3. Any character acceptable
4. String must end in ***an***

Wildcard characters

```
ram: any of various devices for battering, crushing, driving, or forcing something, especially a battering ram
ban: to prevent or forbid such as an event or practice
ear: human organ for hearing
rat: any of several long-tailed rodents of the family Muridae, of the genus Rattus and related genera, distinguished from the mouse by being larger
tan: a colour or to darken with sunlight
ran: simple past tense of run
rap: to strike, especially with a quick, smart, or light blow
car: a form of motor vehicle for personal transport
raw: not having undergone processes of preparing, dressing or manufacture
bar: a long, cylindrical object used for a wide range of p
ebb: to fade away, to recede
```

```
#!/bin/bash

grep -i '^.an' wild1.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge4.sh
ban: to prevent or forbid such as an event or practice
tan: a colour or to darken with sunlight
ran: simple past tense of run
```

Classed wildcards

- Square brackets [ ] are used to restrict a wildcard to be only one of a set of values
- In this example, find lines contain a string starting with R/r followed by any single instance of a *vowel* and ending with **d**

```
Rob
rod
red
ran
RHEL
Rib
rat
rad
rid
rack
```

```
#!/bin/b
grep -i 'r[aeiou]d' wild2.txt
```

1 2 3 4

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge5.sh
rod
red
rad
rid
```

1. Make match case insensitive
2. String must start with R/r
3. Followed by any single vowel instance
4. String must end in a ***d***

Specify allowable range with [ ]

- Square brackets **[ ]** can also specify a range of allowable potential characters
- A string example would be `grep “[A-Z]” text.txt`, i.e look for lines that contain capital letters from A to Z inclusive
- A numeric example would be `grep “[0-9]” text.txt`, i.e look for lines that contain a number

[ ] example - string

```
The organisation conducted the survey in 2017 in response to a gap in the
literature about how professionals in this sector were using digital technology for their work
This study was primarily intended to inform the development of CFCA publications and resources
for the sector. An earlier version of these findings was presented at the Family and Relationship
Services Australia (FRSA) conference in November 2017.
Australian families are increasingly using the internet to procure goods and services but
anecdotal reports suggest social services have been slow to take up digital technology
Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
of Australians were internet users, with this figure highest in the 15-17 years age gro
and lowest in the over-65 years age group (51%) (ABS, 2016). People aged 15-17 yea the
most time online, with an average
Of households with children under
internet-connected devices in eac
users had used the internet to pu
2016).The rates of internet use a
```

Find all lines that start with a capital letter between A and Z inclusive

```
#!/bin/bash

grep '^[A-Z]' art.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge6.sh
The organisation conducted the survey in 2017 in response to a gap in the
This study was primarily intended to inform the development of CFCA publications and resources
Services Australia (FRSA) conference in November 2017.
Australian families are increasingly using the internet to procure goods and services but
Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
Of households with children under 15, 97% had access to the internet, with an average of seven
```

[ ] example - numeric

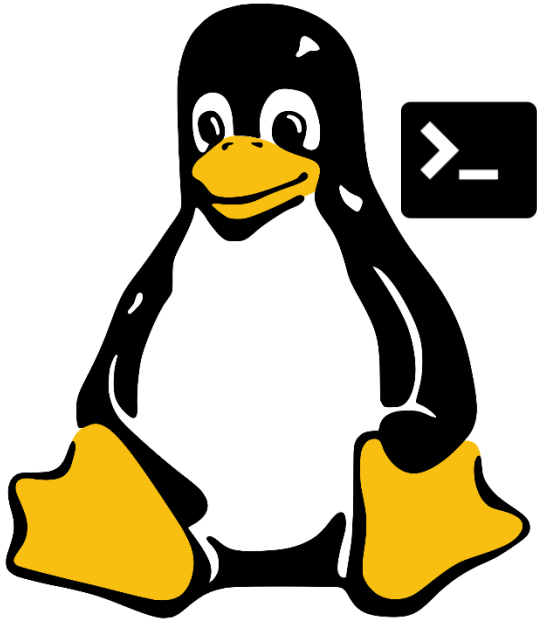
```
The organisation conducted the survey in 2017 in response to a gap in the
literature about how professionals in this sector were using digital technology for their work
This study was primarily intended to inform the development of CFCA publications and resources
for the sector. An earlier version of these findings was presented at the Family and Relationship
Services Australia (FRSA) conference in November 2017.
Australian families are increasingly using the internet to procure goods and services but
anecdotal reports suggest social services have been slow to take up digital technology
Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
of Australians were internet users, with thi
and lowest in the over-65 years age group (5
most time online, with an average of 18 hour
Of households with children under 15, 97% ha
internet-connected devices in each household
users had used the internet to purchase or c
2016).The rates of internet use among Austra
```

Find all lines that a number

```
#!/bin/bash

grep '[1-9]' art.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203./workshop/week4$ ./ge7.sh
The organisation conducted the survey in 2017 in response to a gap in the
Services Australia (FRSA) conference in November 2017.
Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
of Australians were internet users, with this figure highest in the 15-17 years age group (99%),
and lowest in the over-65 years age group (51%) (ABS, 2016). People aged 15-17 years spent the
most time online, with an average of 18 hours spent online for personal use each week (ABS, 2016).
Of households with children under 15, 97% had access to the internet, with an average of seven
internet-connected devices in each household (ABS, 2016). Almost two-thirds (61%) of all internet
2016).The rates of internet use among Australians have been consistently increasing.
```



Extended RegEx Engine

Extended Regex

- ERE can also match with several other collections of classes

Pattern	Effect
[[:alpha:]]	Alphabetical character A-z, a-z
[[:alnum:]]	Alphanumeric character A-z, a-z, 0-9
[[:digit:]]	Digit 0-9
[[:upper:]]	Uppercase A-Z
[[:lower:]]	Lowercase a-z
[[:space:]]	Any whitespace character (space tab newline)
[[:blank:]]	Space or tab
[[:punct:]]	Punctuation character e.g. “!,.;”

Example ERE class - [[:digit:]]

```
#!/bin/bash

grep -e '[[:digit:]]' art.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge10.sh
The organisation conducted the survey in 2017 in response to a gap in the
Services Australia (FRSA) conference in November 2017.
Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
of Australians were internet users, with this figure highest in the 15-17 years age group (99%),
and lowest in the over-65 years age group (51%) (ABS, 2016). People aged 15-17 years spent the
most time online, with an average of 18 hours spent online for personal use each week (ABS, 2016).
Of households with children under 15, 97% had access to the internet, with an average of seven
internet-connected devices in each household (ABS, 2016). Almost two-thirds (61%) of all internet
2016).The rates of internet use among Australians have been consistently increasing.
```

Example ERE class - [[:upper:]]

```
#!/bin/bash

grep -e '^[[:upper:]]' art.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

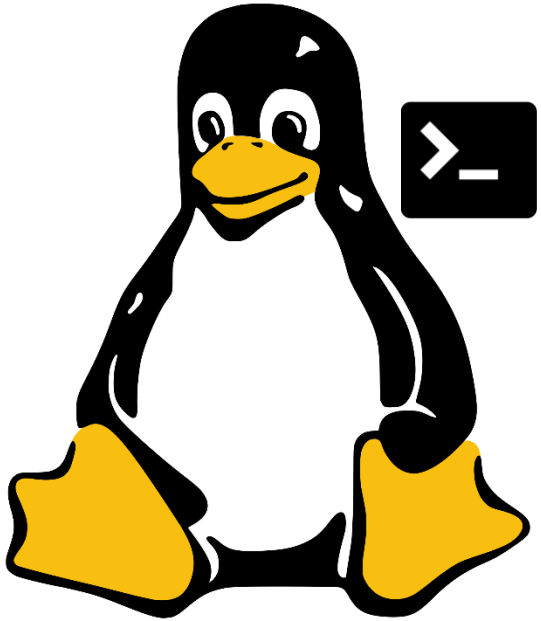
```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge10.sh
The organisation conducted the survey in 2017 in response to a gap in the
This study was primarily intended to inform the development of CFCA publications and resources
Services Australia (FRSA) conference in November 2017.
Australian families are increasingly using the internet to procure goods and services but
Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
Of households with children under 15, 97% had access to the internet, with an average of seven
```

Example ERE class - [[:punct:]]

```
#!/bin/bash

grep -e '[[:punct:]]' art.txt
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge10.sh
for the sector. An earlier version of these findings was presented at the Family and Relationship
Services Australia (FRSA) conference in November 2017.
Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
of Australians were internet users, with this figure highest in the 15-17 years age group (99%),
and lowest in the over-65 years age group (51%) (ABS, 2016). People aged 15-17 years spent the
most time online, with an average of 18 hours spent online for personal use each week (ABS, 2016).
Of households with children under 15, 97% had access to the internet, with an average of seven
internet-connected devices in each household (ABS, 2016). Almost two-thirds (61%) of all internet
users had used the internet to purchase or order goods or services in the last three months (ABS,
2016).The rates of internet use among Australians have been consistently increasing.
```



ERE Repetition and Optionality

The Asterisk Wildcard

- In regex, the asterisk can be used to repeat the previous part of the pattern **0 or more** times
- For example, the grep regex on the right Would find the strings *yes*, *yees* and *yeees*

```
#!/bin/bash

grep 'ye*s' text.txt
```

The Asterisk Wildcard

- The asterisk wildcard can also be used with other regex characters too
- For example, the grep pattern search to the right would find the strings *yes*, *yees*, *yas* and *yaas*

```
#!/bin/bash

grep 'y[ea]*s' text.txt
```

ERE

In the ERE syntax, there are even more useful and versatile pattern matching operators including:

- +
- ?
- {}
- |
- ()

ERE Plus +

- The Plus character "**+**" acts similarly to the asterisk "*****" except instead of *0 or more* repetitions, there must be at least **one or more** repetitions
- For example, the grep pattern search to the right would return the strings *yes*, *yees*, *yas* and *yaas*, but <u>not</u> *ys*

```
#!/bin/bash

grep 'y[ea]\+s' text.txt
```

Question Mark ?

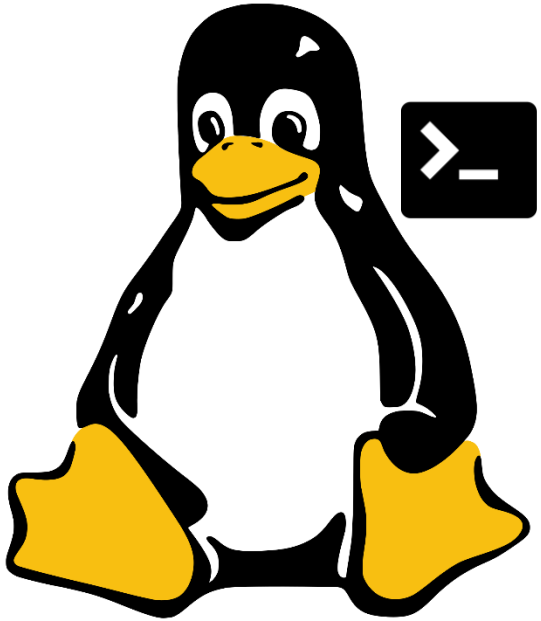
- The question mark character "**?**" acts as an optionality operator, meaning that the preceding character may or may not be present in the pattern being sought
- For example, the grep search to the right would find both *bash* and *ash*

```
#!/bin/bash

grep 'b?ash' text.txt
```

Curly Braces { }

- Braces **{ }** are used to specify a specific number of repetitions of a character or sequence
- For example, the grep pattern search to the right will return *Robert*, *Rabbit* and *Rupert*, but <u>not</u> *Rivet*

```
#!/bin/bash

grep 'R.{4}t' text.txt
```



OR and Expression Grouping

Expression Grouping

- Using parentheses **( )** regex patterns can be grouped together to allow for more complex search patterns to be constructed
- In the example below, **(very)** must be present *positionally* at least once and the **+** indicates what follows must also be *positionally* present
- **Note:** `grep -E` *escapes* traditional usage of special characters. For example, the command `grep -E '{1'` searches for the two-character string `{1` instead of reporting a syntax error in the regular expression

```
#!/bin/bash

grep -E '^regex can be (very)+ confusing' text.txt
```

Expression Grouping

```bash
#!/bin/bash

grep -E '^regex can be (very)+ confusing' text.txt
```

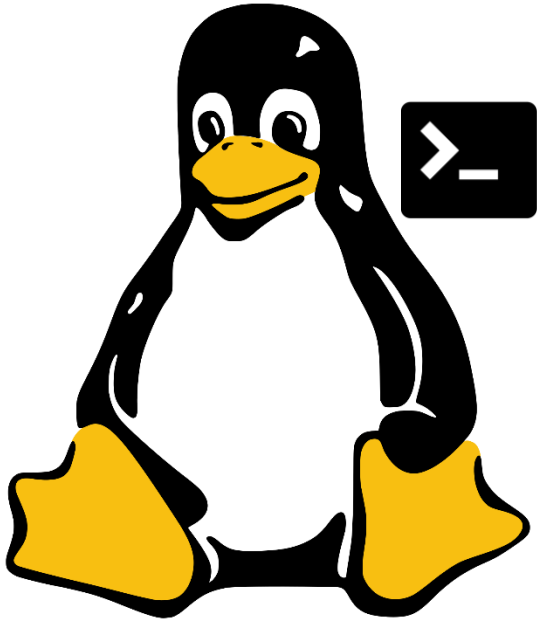
Potential Matches:

```
regex can be very confusing
regex can be very very confusing
regex can be very very very very confusing
```

OR |

- In bash, the pipe operator "**|**" is usually used to redirecting the output of one script or command to the input of another
- Within the context of a regular expression however, it takes on the functionality of **or**
- For example, in the grep pattern search to the right, lines will be returned that <u>end</u> with either the string *bash* or the string *fish*

```
#!/bin/bash

grep -E '(bash$)|(fish$)' text.txt
```



grep options

Common grep options

Option	Description
-c	Suppress normal output; instead print a count of matching lines for each input file
-E	Interpret PATTERN as an extended regular expression.
-i	Ignore case distinctions in both the PATTERN and the input files.
-m *NUM*	Stop reading a file after NUM matching lines.
-n	Prefix each line of output with the line number within its input file.
-o	Show only the part of a matching line that matches PATTERN.
-v	Invert the sense of matching, to select non-matching lines
-w	Select only those lines containing matches that form whole words

grep Options Example

```
1   The organisation conducted the survey in 2017 in response to a gap in the
2   literature about how professionals in this sector were using digital technology for their work
3   This study was primarily intended to inform the development of CFCA publications and resources
4   for the sector. An earlier version of these findings was presented at the Family and Relationship
5   Services Australia (FRSA) conference in November 2017.
6   Australian families are increasingly using the internet to procure goods and services but
7   anecdotal reports suggest social services have been slow to take up digital technology
8   Australian Bureau of Statistics (ABS) research conducted during 2014 and 2015 found that 85%
9   of Australians were internet users, with this figure highest in the 15-17 years age group (99%),
10  and lowest in the over-65 years age group (51%) (ABS,
11  most time online, with an average of 18 hours spent on
12  Of households with children under 15, 97% had access t
13  internet connected devices in each household (ABS, 201
14  users had used the internet to purchase or order goods
15  2016).The rates of internet use among Australians have
```

```
1   #!/bin/bash
2
3   grep -wci 'internet' art.txt
```

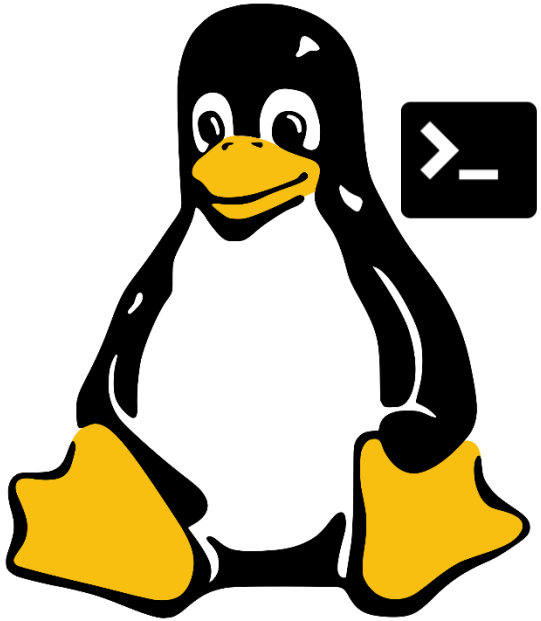
Make case insensitive

Whole word matches only

Count matching lines

DEBUG CONSOLE PROBLEMS

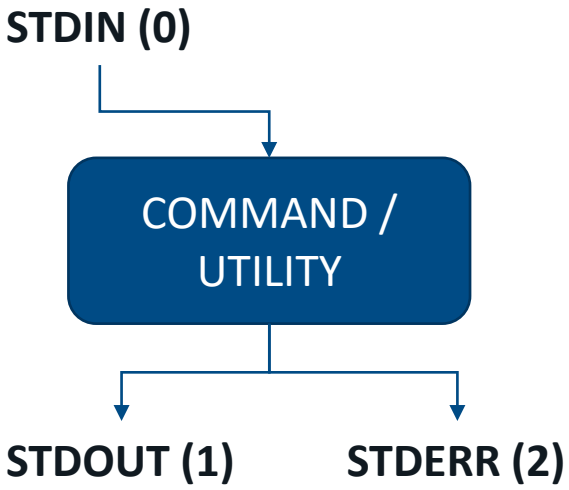
```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/workshop/week4$ ./ge10.sh
6
```



Piping and Redirection

Bash Data Streams

- As discussed in Module 2, bash commands and utilities have automatic access to three (3) data streams:
  - **STDIN (0)** - Standard Input, this is the stream that feeds data into a command or utility
  - **STDOUT (1)** - Standard Output, this is the stream that outputs data from the command or utility; the terminal by default
  - **STDERR (2)** - Standard Error, this is for error messages; also defaults to the terminal)

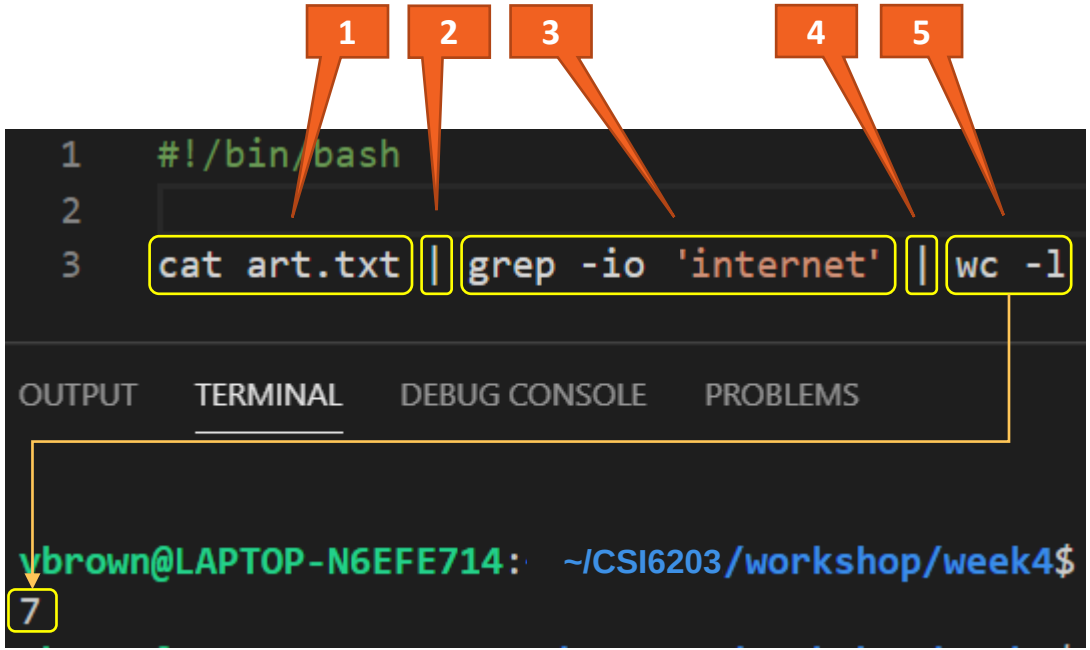


Bash Data Streams

- Bash **pipes** and **redirection** allows streams between command, utilities and files to be connected in specific sequence to manipulate data in useful and flexible ways
- In the example below, three (3) commands are used to count the number of instances of a whole string in a file

```
#!/bin/bash

cat art.txt | grep -io 'internet' | wc -l
```

Example Deconstructed



1	The **cat** command makes a copy of the file named *art.txt*
2	Pipes the data acquired by **cat** to the **STDIN** of the next command (*grep*)
3	The **grep** command retrieves lines of data it receives for each instance of the whole string *internet* in a case-insensitive mode
4	Pipes the data acquired by *grep* to the **STDIN** of the next command (*Word Count*)
5	The Word Count command will count all instances of the string *internet* in the data received line by line

Redirection

- The standard input and standard output can be redirected to use files instead using the redirection operators **<** and **>**

EXAMPLE 1

```
#!/bin/bash

cat art.txt | grep -i 'internet' > output.txt
```

Output the results produced by cat and grep to a file named *output.txt*. If the file does not exist, it will be created.

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714  ~/CSI6203/workshop/week4$ ./ge10.sh
vbrown@LAPTOP-N6EFE714  ~/CSI6203/workshop/week4$ cat output.txt
Australian families are increasingly using the internet to procure goods and services but
of Australians were internet users, with this figure highest in the 15-17 years age group (99%),
Of households with children under 15, 97% had access to the internet with an average of seven
internet connected devices in each household (ABS, 2016). Almost two-thirds (61%) of all internet
users had used the internet to purchase or order goods or services in the last three months (ABS,
2016).The rates of internet use among Australians have been consistently increasing.
```

Redirection Example 2

```
1   #!/bin/bash
2
3   grep -i 'internet' < art.txt > output.txt
```

Output the processed data from grep to a file named *output.txt*. If the file does nor exist, create it.

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
vbrown@LAPTOP-N6EFE714: ~/CSI6203/worksho
vbrown@LAPTOP-N6EFE714: ~/CSI6203/worksho
Australian families are increasingly usi                                  and services but
of Australians were internet users, with                                  years age group (99%),
Of households with children under 15, 97% had access to the internet with an average of seven
internet connected devices in each household (ABS, 2016). Almost two-thirds (61%) of all internet
users had used the internet to purchase or order goods or services in the last three months (ABS,
2016).The rates of internet use among Australians have been consistently increasing.
```

Input the data that grep is to operate on from the file named *art.txt*.

Summary

Terms to Know

- Regular Expressions
- Regular Expression Engines
- Grep and Regex
- Anchors and Wildcards
- Extended RegEx Engine
- ERE Repetition and Optionality
- OR and Expression Grouping
- Common Grep Options
- Piping and Redirection

References and Further Reading

- **Ebrahim, M. & Mallet, A. (2018). *Mastering Linux Based Scripting* (2nd Ed), Chapter 11, pp 194-215.**
- http://regular-expressions.info/engine.html
- http://tldp.org/LDP/Bash-Beginners-Guide/html/chap_04.html
- https://regexr.com/
- https://regex101.com/