Data Engineering COMP2031/8031

Scripting– Week 1

Dr Mehwish Nasim – 2022(S1)

Flinders UNIVERSITY

AWK

- Interpreted programming language designed for text processing
- Name is derived from the family names of its authors – **Alfred Aho, Peter Weinberger, and Brian Kernighan**

- Typical uses of AWK
  - Text processing,
  - Producing formatted text reports,
  - Performing arithmetic operations,
  - Performing string operations, and many more.

AWK

- Linux / mac

Basic Syntax

- awk [options] file ...

- Consider the following example

Coins.txt

```
gold     1    1986  USA                American Eagle
gold     1    1908  Austria-Hungary    Franz Josef 100 Korona
silver  10    1981  USA                ingot
gold     1    1984  Switzerland        ingot
gold     1    1979  RSA                Krugerrand
gold     0.5  1981  RSA                Krugerrand
gold     0.1  1986  PRC                Panda
silver   1    1986  USA                Liberty dollar
gold     0.25 1986  USA                Liberty 5-dollar piece
silver   0.5  1986  USA                Liberty 50-cent piece
silver   1    1987  USA                Constitution dollar
gold     0.25 1987  USA                Constitution 5-dollar piece
gold     1    1988  Canada             Maple Leaf
```

print

- awk '{print $3 "\t" $4}’ coins.txt

```
1986    USA
1908    Austria-Hungary
1981    USA
1984    Switzerland
1979    RSA
1981    RSA
1986    PRC
1986    USA
1986    USA
1986    USA
1987    USA
1987    USA
1988    Canada
```

Pattern matching

- awk '/C/ {print $0}' coins.txt

```
gold      0.1   1986   PRC                  Panda
silver    1     1987   USA                  Constitution dollar
gold      0.25  1987   USA                  Constitution 5-dollar piece
gold      1     1988   Canada               Maple Leaf
```

Pattern matching

- awk '/C/{++cnt} END {print "Count = ", cnt}' coins.txt

- Count = 4

lines that contain more than 50 characters

- 'length($0) > 50' coins.txt

```
gold      1     1986  USA                  American Eagle
gold      1     1908  Austria-Hungary      Franz Josef 100 Korona
silver    1     1986  USA                  Liberty dollar
gold      0.25  1986  USA                  Liberty 5-dollar piece
silver    0.5   1986  USA                  Liberty 50-cent piece
silver    1     1987  USA                  Constitution dollar
gold      0.25  1987  USA                  Constitution 5-dollar piece
```

Flinders UNIVERSITY

- The following special variables are builtin in awk:
- FS : acts as field separator to splits awk input lines in fields. I can be a single character, FS="c" ; a null string, FS="" (then each individual character becomes a separate field); a regular expression without slashes, FS="re" ; FS=" " stands for runs of spaces and tabs and is defaults value.
- NF : the number of fields to read;

- $1 , $2 , ...: 1st field, 2nd field. etc. of the current input line,
- $0 : current input line;
- NR : current put line number.
- OFS : string to collate fields when printed.
- ORS : output record separator, by default a newline.
- RS : Input line (record) separator. Defaults to newline. Set as FS .
- IGNORECASE : affects FS and RS when are regular expression;

regexp

- awk 'BEGIN {print "Coins"}
- /gold/{i++; print $0}
- END {print i " lines out of " NR}' coins.txt

```
nasi0029@C02FW0D3ML85 Tutorials % awk 'BEGIN {print "Coins"} /gold/{i++; print $
0}  END {print i " lines out of " NR}' coins.txt
Coins
gold     1    1986  USA                American Eagle
gold     1    1908  Austria-Hungary    Franz Josef 100 Korona
gold     1    1984  Switzerland        ingot
gold     1    1979  RSA                Krugerrand
gold     0.5  1981  RSA                Krugerrand
gold     0.1  1986  PRC                Panda
gold     0.25 1986  USA                Liberty 5-dollar piece
gold     0.25 1987  USA                Constitution 5-dollar piece
gold     1    1988  Canada             Maple Leaf
9 lines out of 13
```

AWK – minimal theory

- 'BEGIN {<init actions>};
- <cond1> {<program actions>};
- <cond2> {<program actions>};
- ...
- END {<final actions>}'

- awk 'BEGIN {print "Coins"} /gold/{i++; print $0} END {print i " lines out of " NR}' coins.txt

- awk 'BEGIN {print "First 3 coins"} NR<4' coins.txt
- What is this command going to output?

Conditions

- Exampels
- awk 'NR % 6' # prints all lines except those divisible by 6
- awk 'NR > 5' # prints from line 6 onwards
- awk '$2 == "foo"' # prints lines where the second field is "foo"

Some string functions

- awk '{print substr($3,3) " " substr($4,1,3)}'

Other unix commands

- vi
  - Opens a text editor
  - !q quit without saving
  - !wq quit and save

grep

- grep 'word' filename
- grep -i 'bar' file1
- grep -R 'httpd’ .