AWK cookbook

Nina Dombrowski

```
r Sys.Date()
```

Table of contents

Preface

This is a cook book for key AWK commands.

1 AWK basics

1.1 Formating tables

1.1.1 Replacing empty space with symbols

In a tab delimited file, replace any empty field with a -

```
awk 'BEGIN {FS = OFS = "\t"} {for(i=1; i<=NF; i++) if($i ~ /^ *$/) $i = "-" }; 1' input > input_new
```

1.1.2 Add new columns based on conditions

Example usage: If a value in column 4 is larger than the value in column 5 give it a notification, otherwise use a -.

```
wk  -v OFS='\t' '{if ($4 > $5){ $7="high_score" }else{ $7="-"} print }' input > input_new
```

1.1.3 Replacing column names

Example: replace the first column name to accession

```
awk 'BEGIN{FS="\t";FS="\t"; OFS="\t"}{if(NR==1) $1="accession"} {print $0 }' input > input_new
```

1.2 Subsetting tables

1.2.1 Print lines that have specific pattern.

```
#just print rows if they contain the word hello in the 5th column
awk  '$5 == "hello"' Input

#look for a pattern and define what we want to print
awk '/hello/ {print $1}' Input
awk '$5 == "hello" {print $1}' Input

#we can also search for more than one pattern
awk '$1 ~ /John|Eric/ {print $0}' Input

#print everything, except rows with John and Eric
awk '!/John|Eric/ {print $1,$3}' Input
awk '$1 !~ /John|Eric/ {print $1,$3}' Input
```

1.2.2 Split tables by categories

Imagine we have a table listing people from different states, the states are listed in column 3. With awk we can easily split this one table, into multiple tables, one each per state

```
#print all columns, one table/state
awk '{print > $3".txt"}'  Input

#print only the first column, one table/state
awk '{print $1 > $3".txt"}'  Input
```

This command will generate several new text files in our working directory.

1.3 Splitting columns

We can also split columns. For example, we could have something like this in the first column: BinID-accession. Now we want to split this in two columns, one for the binID , the second should contain the accession.

Some new syntax for AWK:

- **split()** = the function 'split' divides a string into pieces
- **$1** = The column we want to split
- **"x"** = The pattern we want to use for splitting
- **a** = We name the ARRAY we generate 'a'. An ARRAY is similar to a variable you just can store more information in the array we store the different pieces that were split
- **a[1]** = retrieve the first part of the split array =E
- **a[2]** = retrieve the second part of the split array =1

```
awk 'BEGIN{FS=OFS="\t"}{split($1,a,"-")} {print $1,a[1],a[2]}' Input
```

1.4 print the header of a table

```
awk -F'\t' '{for (i = 1; i <= NF; ++i) print i, $i; exit }' temp1
```

2 Math with AWK

2.1 Summarizing a column

```
awk -F'\t' -v OFS='\t' '{sum+=$2} END {print sum}'  File.txt
```

3 Dealing with Sequence Data

3.1 Counting the number of sequences/file

```
awk -v OFS='\t' '/>/ {count++} END{print FILENAME, count}' Input.faa
```

3.1.1 Adding the filename into the fasta header

```
for i in *fna; do awk '/>/{sub(">","&"FILENAME"-");sub(/\.fna/,x)}1' $i > renamed/$i; done
```