Coding tasks for 26.11.2020

Topic: Basic file filtering with awk

1. How many records are in the sample-file?

Steps:

→ execute command at the end: END

→ NR stores number of read lines

Solution:

```
awk 'END {print NR}' <file_name>
```

Alternative on the command-line: `wc -l <file_name>`

2. Print the first line of the sample file

Steps:

→ only execute following command for line 1 (NR == 1)

→ the variable $0 contains the full line

Solution:

```
awk 'NR == 1 {print $0}' <file_name>
```

Alternative on the command-line: `head -n 1 <file_name>`

3. Print every second line

Steps:

→ use modulo operator (in awk: %)

→ most calculation operations can be used in awk

→ print command not needed (standard operation)

Solution:

```
awk 'NR%2 == 1' <file_name>
```

4. Get the bitwise flag of "SRR9077711.2552"

Steps:

→ $1 to $n contain the different columns of a line

→ logical operators can be used in awk

Solution:

```
awk '$1 == “SRR9077711.2552“ {print $2}' <file_name>
```

5. What is the mean length of all segment sequences?

Steps:

→ it is possible to set own variables in an awk command

→ variables can be set on the fly

→ there are basic functions in awk like length()

Solution:

```
awk '{sum += length($10)} END {print sum/NR}' <file_name>
```

6. What is the maximum length of all segment sequences?

Steps:

→ if, for and while are also available

→ combination of logical and calculation operations can be used for extensive numerical filtering

Solution:

```
awk 'BEGIN {max = 0} {if (length($10) > max) max = length($10)}
END {print max} <file_name>
```

7. Find all mappings that contain the sequence "AGACACCG"

Steps:

→ for filtering strings, regular expressions can be used

→ '~' assigns a line to the regex

→ '/.../' contains the regex

Solution:

```
awk '$10~/(AGACACCG)/' <file_name>
```

8. Get all mappings to chr9 or chr7 with a segment sequence length of 71-73 and edit-distance of 0

Steps:

→ for more specific searches, regex can be combined

→ the combination of numerical and regex filtering can be used for even very complex filtering scripts

Solution:

```
awk '$3~/(chr9)|(chr7)/ && length($10)~/[71-73]/ &&
$0~/(NM:i:0)/' <file_name>
```

9. Write out read, chromosome and CIGAR-string for all mappings into a csv-file

Steps:

→ the output of awk can be customised

→ output field separator (OFS) sets the separator between columns of a line

→ output record separator (ORS) sets the separator between the lines

→ similar, FS and RS can be set for different input separators

Solution:

```
awk -v OFS=“,“ '{print $1, $3, $6}' <file_name>
```

Conclusion

→ awk can be a useful tool for filtering files and generate specific output-formates

→ faster then normal shell-commands

→ very compact and simple (most of the time)

→ easy usage allows throwaway-commands

→ can be used on most platforms (Linux, Windows, ...)