Coding tasks for 31.05.2021

<u>Topic: Basic shell tools for viewing and modifying data</u>

I. Introduction:

There are several tools for looking at and modifying data on the shell directly. Also most analysis is done elsewhere, it can be useful to know some of these. The advantage of them is a relative easy and fast applicability, which can help in many different situations. Especially with larger files, filtering would not be possible in any other way or would be much more time-consuming. The goal for this task therefore is to take a look at some of these tools and find out how to use them.

1. The first look at a file: Head, tail and less

No matter if working with a previous unknown file format or with a known one, it can be useful to take a first look at the data via the shell. For example a blank line in a FASTA file can lead to crashes of analysis or assembly tools. To prevent this head or tail can be used to look at the few first and last lines of a file. For a more in-depth look at the data less can be used.

2. Get specific information: Grep

If looking for a specific mapping or string grep is the way to go. It is simple to use and can help to easily find patterns or entries, even in files with a large size.

```
grep [option...] [patterns] [file...]
```

3. Filtering a file: Awk

AWK is a commonly used tool for filtering and extracting specific data. But not only that, to be clear AWK is a computer language on its own and was released in 1977.
Small awk-commands are a easy method to work with files on the command-line or to perform some fast filtering before using the files elsewhere. Of course it can also be used to write longer scripts for all sorts of more complex filtering or data extraction.
Using awk as a command-line tool looks like this:

```
awk ' condition { action }' file

awk ' $2==0 { print $1 }' coding_task_sample.sam
```

4. Editing a file: Sed

As AWK is mostly used for filtering the data, sed it the tool for editing it in different ways. The

most common usage of sed is to replace or delete given strings in a file. This can be useful to change some formats, header or remove errors. Combined with regex sed can be used as a fast option to change complete files. But that would be too much for these tasks.
Using sed on the command line looks like this:

```
sed s/pattern1/pattern2/
```

This command replaces the first occurrence of the first pattern with the second one.

II. Tasks:

There are hints for some task and useful links at the end. All tasks should be solved on the coding_task_sample.sam-file and can be done by looking into the manual of the tools.

Head, tail and less:

1. Check head and tail on the sam-file. Is there anything unusual?

2. How many lines are printed and how can the amount of lines be changed?

Grep:

1. To which chromosome has the read SRR9077711.3518 mapped?

2. Can grep print more then one line after the found string? If yes how?
Additional info: This can also be useful for searching in FASTA and FASTQ files, where each entry consists of two or four lines.

3. How much mappings have an edit-distance of 0?
Hint: The edit-distance is noted as NM:i:<distance> in the sam-file.

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

AWK:

1. How many records are in the sample-file?
Hint: NR contains number of line, print the one at the END.

2. Print the first line of the sample file.
Hint: $0 contains the full line, print the one at NR = 1.

3. Print every second line.
Hint: Use modulo on number of line (NR%2).

4. Get the bitwise flag of "SRR9077711.3997".
Hint: Check if first column ($1) contains the string, then print the bitwise flag ($2).

5. What is the mean length of all segment sequences?
Hint: You can sum up columns (var += $10), get length of the string first (length($10)).

6. What is the maximum length of all segment sequences?
Hint: If clauses can be used in awk ({if ($2 > x) x=$2}).

Sed:

1. Remove all mappings to non conventional chromosomes.
Hint: Keep all mappings which contain "	chr" as string. Don't forget the tab.

2. Globally change SRR9077711 to SRR9077715.

3. Replace all appearances of "--" with nothing.

Final task:

Remove all blank lines with awk or sed and write the result in a new file.

III. Useful links for the tasks:

SAM file format:	https://en.wikipedia.org/wiki/SAM_(file_format)
Small regex-cheat-sheet:	https://tinyurl.com/y62lmdl3
Small awk cheat sheet:	https://www.shortcutfoo.com/app/dojos/awk/cheatsheet
Sed manual page:	https://www.gnu.org/software/sed/manual/sed.html
Grep manual page:	https://www.gnu.org/software/grep/manual/grep.html