Bash Shell Scripting

cut command

```
#!/bin/bash
```

Learn how to automate common tasks using bash shell scripting

Cut command:

- The 'cut‘ command is a powerful tool to extract parts of each line from a file.
- It is based on
  - Byte Position
  - Character Position
  - Fields based on delimiter (by default delimiter is the tab)

- Cut command syntax:
  - cut [options] <positions(fields) /range of positions(fields)> <input_file>
  - cat file | cut [options] <positions(fields) /range of positions(fields)>
    - Options: -b -c and -f
    - Rages:

      2 only second byte/character/filed

      2- second byte/character/filed to last

      -7 first to seven

      3,5 third and fifth

Cut command for Byte/Character Position:

- To cut out a section of a line by specifying a byte/character position use the -b/-c option.
- Syntax:
  - cut -b <position's/range of position's> file
  - cut -c <position's/range of position's> file
  - Position's: 3,5,10
  - Range of Position's: 3-7, 6-10
- Ex: mytext.txt
- cut -b 2 mytext.txt
- cut -b 3,7 mytext.txt
- cut -b 5-9 mytext.txt
- cut -b 5- mytext.txt
- cut -b -7, 9 mytext.txt
- Use --complement to complement the output

Cut command for filed Position:

- To cut out a section of a line by specifying a field position use the -f option.
- Assume fields are like columns, by default cut command will separates columns based on tab(delimiter).
- If we want to use different filed separator use -d (delimiter).
- Syntax:
  - cut -f <position's/range of position's> file
  - cut -f <position's/range of position'ss> [-d ':'] [--output-delimiter='**'] file
  - -d is a delimiter like @ , : / etc….
  - Position's: *3,5,2*
  - Range of Position's: *3-7, 6-10*
- Ex: mytext.txt
- cut -f *2* mytext.txt
- cut -f *3,7* mytext.txt
- cut -f *5-9* mytext.txt
- cut -f *5-* mytext.txt
- cut -f -7, 9 --output-delimiter=" " mytext.txt

Use -s option with –f to Ignore the line that do not contain a delimiter

Thank you