**Text Processing**

1. **Sed**

**SED command in UNIX stands for stream editor and it can perform lots of functions on file like searching, find and replace, insertion or deletion. Though most common use of SED command in UNIX is for substitution or for find and replace. By using SED you can edit files even without opening them, which is much quicker way to find and replace something in file, than first opening that file in VI Editor and then changing it.**

* **SED is a powerful text stream editor. Can do insertion, deletion, search and replace(substitution).**

**Sample file**

**cat > repeating.txt << EOL**

**apple apple orange banana**

**banana banana banana apple**

**grape grape grape grape**

**orange orange orange orange**

**banana banana apple apple**

**EOL**

**SED command in unix supports regular expression which allows it perform complex pattern matching.**

**Replacing or substituting string : Sed command is mostly used to replace the text in a file. The below simple sed command replaces the word “text1” with “text2” in the file.**

**cat > demo.txt**

**sed 's/tex1/tex2' demo.txt**

**Here the “s” specifies the substitution operation. The “/” are delimiters. The “text1” is the search pattern and the “text2” is the replacement string.**

**By default, the sed command replaces the first occurrence of the pattern in each line and it won’t replace the second, third…occurrence in the line.**

**Replacing the nth occurrence of a pattern in a line : Use the /1, /2 etc flags to replace the first, second occurrence of a pattern in a line. The below command replaces the second occurrence of the word “text1” with “text2” in a line.**

**sed 's/text1/text2/2' demo.txt**

**here, 2 is used for 2nd occurence.**

**Replacing all the occurrence of the pattern in a line : The substitute flag /g (global replacement) specifies the sed command to replace all the occurrences of the string in the line.**

**sed 's/text1/text2/g' demo.txt**

**Replacing from nth occurrence to all occurrences in a line : Use the combination of /1, /2 etc and /g to replace all the patterns from the nth occurrence of a pattern in a line. The following sed command replaces the third, fourth, fifth… “text1” word with “text2” word in a line.**

**sed 's/text1/text2/2g' demo.txt**

**Replacing string on a specific line number : You can restrict the sed command to replace the string on a specific line number. An example is**

**sed '3 s/text1/text2' demo.txt**

**this will changed in 3rd line.**

**Replacing string on a range of lines : You can specify a range of line numbers to the sed command for replacing a string**

**sed '1,3 s/text1/text2/' demo.txt**

**Here the sed command replaces the lines with range from 1 to 3.**

**sed '2,$ s/text1/text2' demo.txt**

**Here $ indicates the last line in the file. So the sed command replaces the text from second line to last line in the file.**

**Deleting lines from a particular file : SED command can also be used for deleting lines from a particular file. SED command is used for performing deletion operation without even opening the file**

**1. To Delete a particular line say n in this example**

**sed 'nd' demo.txt**

**Example:**

**sed '5d' demo.txt**

**2. to delete last last line**

**sed '$d' demo.txt**

**3. To Delete line from range x to y**

**sed 'x, yd' demo.txt**

**example:**

**sed '3,5d' demo.txt**

**4. To Delete from nth to last line**

**sed 'n, $d' demo.txt**

**example:**

**sed '3,$d' demo.txt**

**5. To Delete pattern matching line**

**Syntax:**

**$ sed '/pattern/d' demo.txt**

**Example:**

**$ sed '/abc/d' demo.txt**

**Question :-**

**To remove all occurrences of the # character from a file using sed**

**To replace the URL https://amazon.com with www.amazon.com in a file using sed, you can use the following command:**

**sed 's|https://amazon.com|www.amazon.com|g' filename.txt**

**Here’s a breakdown of the command:**

* **s|pattern|replacement|g: This tells sed to substitute (s) the pattern with the replacement globally (g) in each line.**
* **|: The delimiter can be any character. Here, | is used to avoid conflict with the slashes in the URL.**

### **Example**

**Let's assume you have a file called urls.txt with the following content:**

**Visit our website at https://amazon.com for more information.**

**Our affiliate link is https://amazon.com/affiliate.**

**For secure shopping, go to https://amazon.com.**

**To replace https://amazon.com with www.amazon.com, you would use:**

**sed 's|https://amazon.com|www.amazon.com|g' urls.txt**

**Output:**

**Visit our website at www.amazon.com for more information.**

**Our affiliate link is www.amazon.com/affiliate.**

**For secure shopping, go to www.amazon.com.**

### **Saving Changes to a File**

**To save the changes back to the original file, use the -i option with sed:**

**sed -i 's|https://amazon.com|www.amazon.com|g' urls.txt**

**This command modifies the urls.txt file in place, replacing all occurrences of https://amazon.com with** [**www.amazon.com**](http://www.amazon.com)**.**

**cat > complex\_example.txt << EOL**

**apple banana**

**apple apple orange banana**

**Visit https://amazon.com for more.**

**file\_path needs to be updated.**

**fruit: apple, vegetable: carrot**

**orange juice is tasty.**

**apple**

**banana**

**grape grape apple**

**EOL**

**Replace "apple" with "orange" only if the line contains "banana"**

**sed '/banana/ s/apple/orange/g' complex\_example.txt**

**Reference :-** [**https://www.linkedin.com/pulse/sed-command-linuxunix-example-anand-mishra-fht0e/**](https://www.linkedin.com/pulse/sed-command-linuxunix-example-anand-mishra-fht0e/)

[**https://phoenixnap.com/kb/linux-sed#ftoc-heading-8**](https://phoenixnap.com/kb/linux-sed#ftoc-heading-8)

1. **awk**

**The awk command in Linux is a powerful text-processing tool that allows you to manipulate and analyze text files and streams. It works by processing each line of input according to a set of rules and patterns specified in a script.**

### **Basic Syntax**

**The basic syntax of awk is:**

**awk 'pattern {action}' file**

### **Examples of awk Usage**

#### **Example 1: Print Specific Columns**

**Consider a file data.txt with the following content:**

**John Doe 30**

**Jane Smith 25**

**Alice Johnson 28**

**To print the first and last names (columns 1 and 2):**

**awk '{print $1, $2}' data.txt**

1. **Cut**

**The cut command in Linux is used to extract specific sections from lines of a file or stream. It can be used to cut sections based on delimiters, character positions, or byte positions.**

### **Basic Syntax**

**The basic syntax of cut is:**

**cut [options] [file]**

### **Options**

1. **-f: Specifies the fields to cut (for fields separated by delimiters).**
2. **-d: Specifies the delimiter character (default is tab).**
3. **-c: Specifies the character positions to cut.**
4. **-b: Specifies the byte positions to cut.**

**4) Unique**

**The uniq command in Linux is used to filter out repeated lines in a file or from the output of another command. It typically works in conjunction with the sort command because uniq only removes consecutive repeated lines.**

### **Example 1: Basic Usage**

**Suppose you have a file called file.txt with the following content:**

**apple**

**banana**

**apple**

**orange**

**banana**

**apple**

**sort file.txt | uniq**

**Output:**

**apple**

**banana**

**orange**

### **Example 2: Counting Repeated Lines**

**To count the number of times each line appears, use the -c option:**

**sort file.txt | uniq -c**

**3 apple**

**2 banana**

**1 orange**

**5) Xargs**

**The xargs command in Linux is used to build and execute command lines from standard input. It is particularly useful for handling the output of other commands and passing it as arguments to another command.**

### **Basic Usage**

**The basic syntax of xargs is:**

**command | xargs [command [initial-arguments]]**

### **Example 1: Removing Files**

**If you have a list of files to delete and you want to pass them to the rm command, you can use xargs:**

**# Create example files**

**touch file1.txt file2.txt file3.txt**

**# List files and delete them using xargs**

**ls \*.txt | xargs rm**

**This will delete all .txt files in the current directory.**

### **Example 2: Counting Lines in Multiple Files**

**Use xargs to count the number of lines in multiple files:**

**# Create example files**

**echo -e "line1\nline2\nline3" > file1.txt**

**echo -e "line1\nline2" > file2.txt**

**# Count lines in files**

**ls \*.txt | xargs wc -l**

**6) set**

**The set command in bash is used to configure the shell environment, manage shell options, and control the behavior of the shell. It can be used to enable or disable options, set positional parameters, and more.**

### **Common Uses of the set Command**

#### **1. Viewing All Shell Variables**

**Running set without any arguments will display all shell variables, including environment variables and shell functions.**

**set**

#### **2. Setting Shell Options**

**You can use set to enable or disable specific shell options. This is done using flags with set.**

* **Enable an Option**

**# Enable debug mode (print commands before executing)**

**set -x**

**Disable an Option**

**# Disable debug mode**

**set +x**

**Enable Exit on Error**

**# Exit the script if any command fails**

**set -e**

**script.sh**

**This script uses set to assign the current date and time to positional parameters and then prints them out.**

**#! /bin/bash**

**# Set the current date and time to positional parameters**

**set -- "$(date "+%Y-%m-%d")" "$(date "+%H:%M:%S")"**

**# Access and display the date and time**

**echo "Current Date: $1"**

**echo "Current Time: $2"**

**Output:-**

**Current Date: 2024-08-03**

**Current Time: 15:35:21**

### **Explanation**

* **set -- "$(date "+%Y-%m-%d")" "$(date "+%H:%M:%S")": This command sets positional parameters $1 and $2 with the current date and time, respectively.**
* **$1: Contains the current date.**
* **$2: Contains the current time.**