|  |  |  |
| --- | --- | --- |
| ACAD-DI-86 | **Lab Manual** | Academic Year: 2023-24 |
| Rev : 01 | Semester: II |
| Date: 02.04.2021 |

**Subject:** Operating Systems Lab (BTCOL406)

**List of Experiments**

|  |  |
| --- | --- |
| **Exp. No.** | **Name of Experiments** |
| 01 | Hands on Unix Commands. |
| 02 | Shell Script programming using the commands grep, awk, and sed. |
| 03 | Implementation of various CPU scheduling algorithms (FCFS, SJF). |
| 04 | Concurrent programming; use of threads and processes, system calls (fork and v-fork). |
| 05 | Study Pthreads and implement the following: write a program which shows a performance. |
| 06 | Implementation of Producer-Consumer problem. |
| 07 | Implementation of various page replacement algorithms (FIFO, LRU). |
| 08 | Implementation of various memory allocation algorithms, (First fit, Best fit and Worst fit). |
| 09 | Implementation of Bankers algorithm. |
| 10 | Scheduling algorithms (FCFS, SCAN, SSTF, C-SCAN). |

**Experiment No: 02**

**Aim:** Shell Script programming using the commands grep, awk, and sed.

**Objectives:** To study and implement various Shell Script programming UNIX Commands like grep, awk, sed.

**Theory:**

Grep, sed, and AWK are all standard Linux tools that are able to process text. Each of these tools can read text files line-by-line and use regular expressions to perform operations on specific parts of the file. Grep is used for finding text patterns in a file and is the simplest of the three. Sed can find and modify data, however, its syntax is a bit more complex than grep. AWK is a full-fledged programming language that can process text and perform comparison and arithmetic operations on the extracted text.

**grep:**

grep is a Linux utility used to find lines of text in files or input streams using regular expressions. It’s name is short for Global Regular Expression Pattern

Beginning at the first line in the file, grep copies a line into a buffer, compares it against the search string, and if the comparison passes, prints the line to the screen. Grep will repeat this process until the Example data file.

Here is the basic syntax of the grep command:

grep [options] pattern [file...]

Ex. Consider a text file ‘good’

goodness

goodlier

goodwills

goodwives

goodies

goodbyes

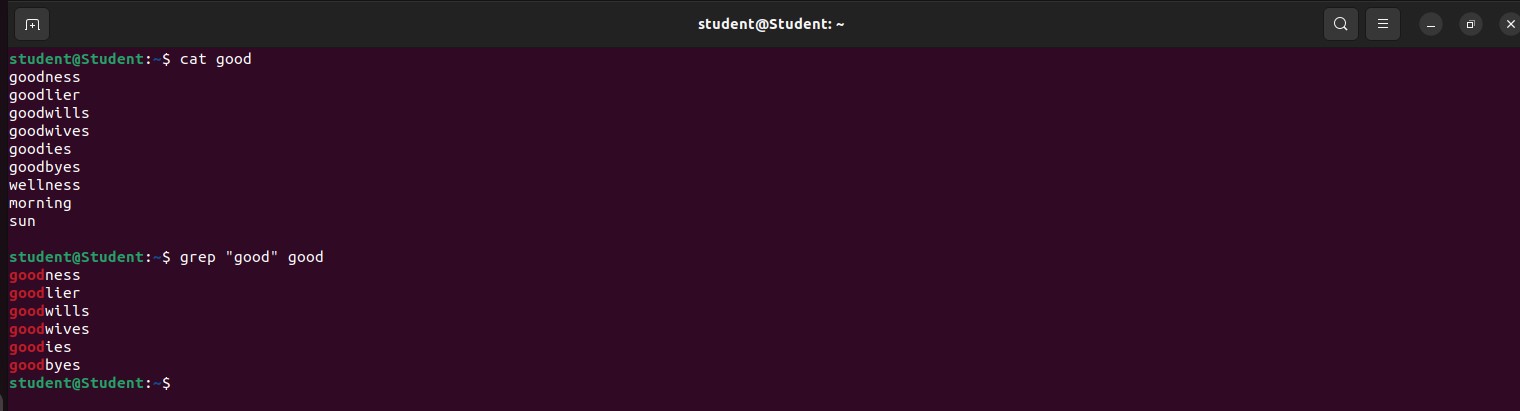
wellness

morning

sun

1. **“pattern”** : To print any line from a file that contains a specific pattern of characters.

$grep “good” good



1. **–n :** When grep prints results with many matches, it comes handy to see the line numbers.

$grep –n “good” good

1. **–vn** : To print any line from a file that doesn’t contain a specific pattern of characters.

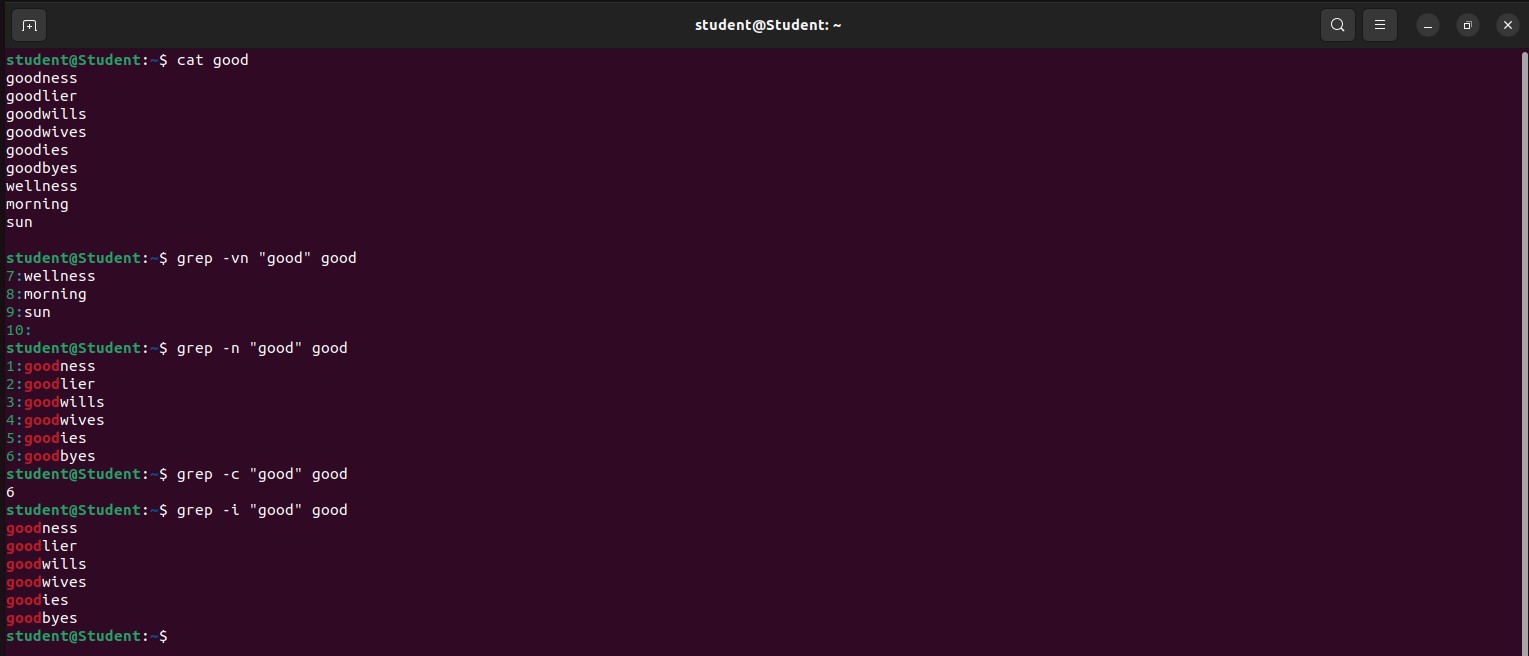
$grep –vn “good” good

1. **–i** : ignores the case of the text string.

$grep –i “good” good

1. **–c :** To print a count of matching lines to standard output.

$grep –c “good” good



**awk:**

Awk is a scripting language used for manipulating data and generating reports. The awk command programming language requires no compiling and allows the user to use variables, numeric functions, string functions, and logical operators.

Awk is a utility that enables a programmer to write tiny but effective programs in the form of statements that define text patterns that are to be searched for in each line of a document and the action that is to be taken when a match is found within a line. Awk is mostly used for pattern scanning and processing. It searches one or more files to see if they contain lines that matches with the specified patterns and then perform the associated actions.

Awk is abbreviated from the names of the developers – Aho, Weinberger, and Kernighan.

awk 'pattern { action }' file

The awk command in Ubuntu (and other Unix-like operating systems) is a versatile text-processing tool used for pattern scanning and processing. It operates on a per-line basis, allowing users to specify patterns and actions to perform on lines that match those patterns. awk derives its name from the initials of its designers: Alfred Aho, Peter Weinberger, and Brian Kernighan.

Here is the basic syntax of the awk command:

awk 'pattern { action }' file

* **pattern:** This is a condition or a regular expression that is tested against each line of input. If the pattern matches, the associated action is executed.
* **action:** This is the action to be performed when the pattern matches. It can be any valid AWK script.
* **file:** This is the input file that awk will process. If not provided, awk reads from standard input.

Consider the following text file as the input file for all cases below:

**$cat > person.txt**

ajay manager account 45000

sunil clerk account 25000

varun manager sales 50000

amit manager account 47000

tarun peon sales 15000

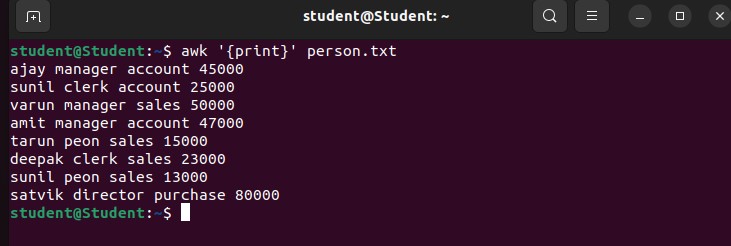
deepak clerk sales 23000

sunil peon sales 13000

satvik director purchase 80000

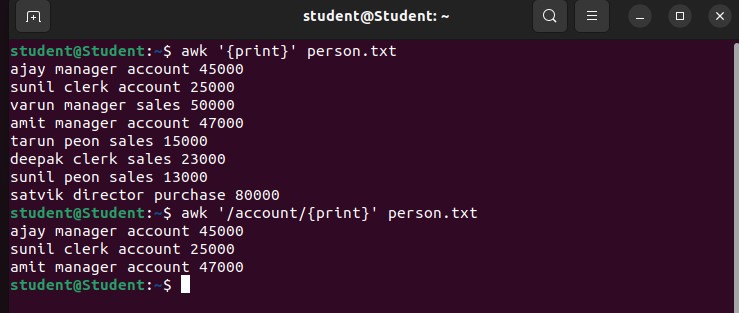
**1. Default behavior of Awk**: By default Awk prints every line of data from the specified file.

$ awk '{print}' person.txt



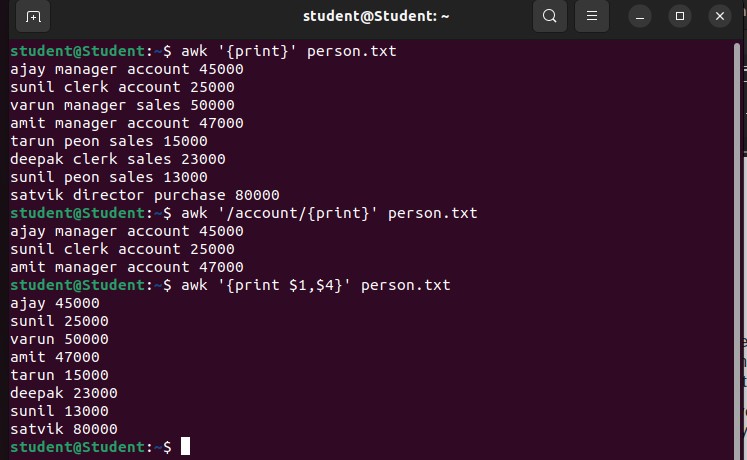
**2. Print the lines which match the given pattern**.

$ awk '/account/ {print}' person.txt



**3. Splitting a Line Into Fields :** For each record i.e line, the awk command splits the record delimited by whitespace character by default and stores it in the $n variables. If the line has 4 words, it will be stored in $1, $2, $3 and $4 respectively. Also, $0 represents the whole line.

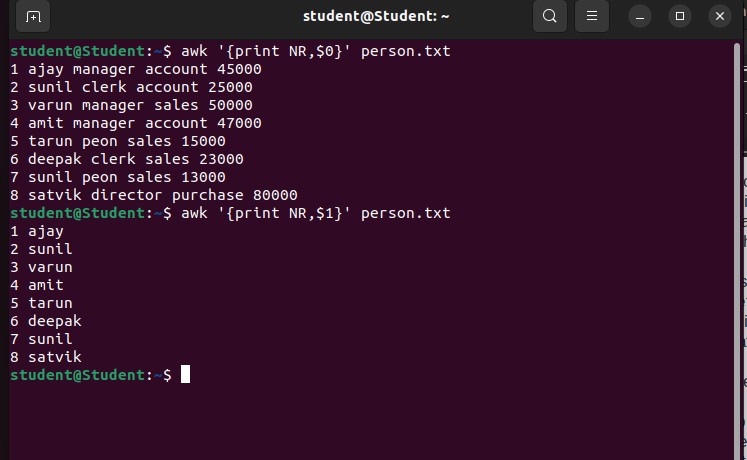
$ awk '{print $1,$4}' person.txt



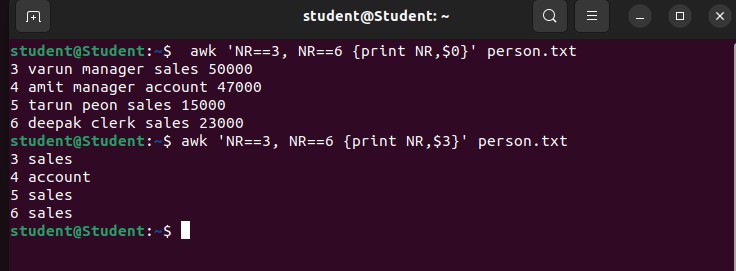
**4. Use of NR built-in variables (Display Line Number)**

$ awk '{print NR,$0}' person.txt

$ awk '{print NR,$1}' person.txt

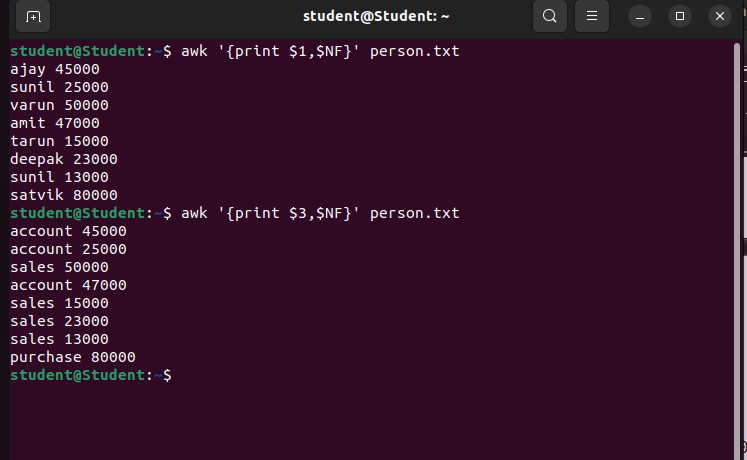


5. **Another use of NR built-in variables (Display Line From 3 to 6)**



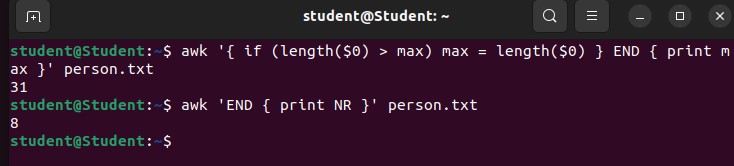
**6 . Use of NF built-in variables (Display Last Field)**

$ awk '{print $1,$NF}' person.txt



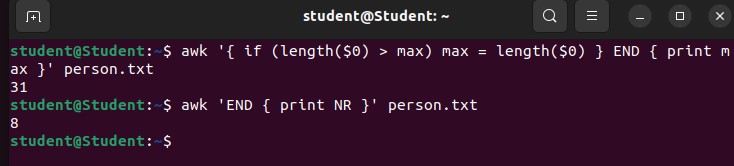
**7. To find the length of the longest line present in the file:**

$ awk '{ if (length($0) > max) max = length($0) } END { print max }' person.txt



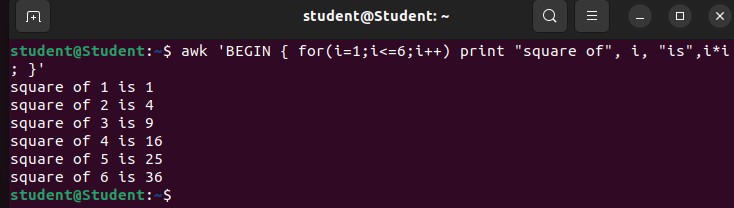
**8. Print the number of lines in a file:**

$ awk 'END {print NR}' person.txt



9. **To print the squares of first numbers from 1 to n say 6:**

$ awk 'BEGIN { for(i=1;i<=6;i++) print "square of", i, "is",i\*i; }'



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

Here is the basic syntax of the sed command:

**sed options... [script] [inputfile...]**

Consider the below text file as an input.

**$cat > geekfile.txt**

unix is great os. unix is opensource. unix is free os.

learn operating system.

unix linux which one you choose.

unix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

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

**$sed 's/unix/linux/' unix**

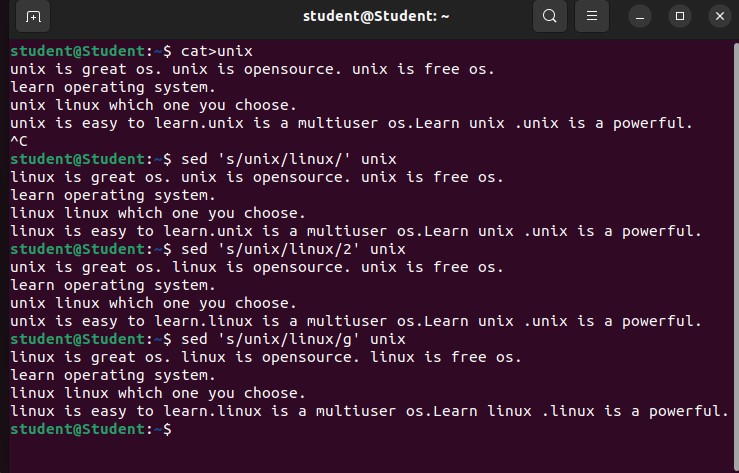
By default, the sed command replaces the first occurrence of the pattern in each line and it won’t replace the second, third…occurrence in the line.

**2. 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 “unix” with “linux” in a line.

**$sed 's/unix/linux/2' unix**

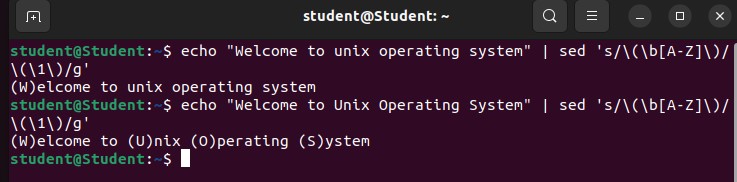
**3. Replacing all the occurrence of the pattern in a line :**The substitute flag /g (global replacement) specifies the sed command to replace all the occurrences of the string in the line.

**$sed 's/unix/linux/g' unix**



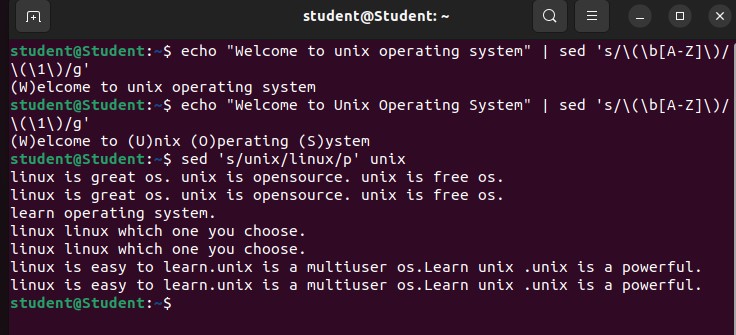
**4. Parenthesize first character of each word :**This sed example prints the first character of every word in parenthesis.

$ echo "Welcome to Unix Operating System" | sed 's/\(\b[A-Z]\)/\(\1\)/g'



**5. Duplicating the replaced line with /p flag :**The /p print flag prints the replaced line twice on the terminal. If a line does not have the search pattern and is not replaced, then the /p prints that line only once.

**$sed 's/unix/linux/p' unix**



**6. 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  
Examples:  
**To Delete a particular line say n in this example**

Syntax:

$ sed 'nd' filename.txt

$ sed '2d' filename.txt

