**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.

# What can we do with awk?

1. awk Operations
   * Scans a file line by line
   * Splits each input line into fields
   * Compares input line/fields to pattern
   * Performs action(s) on matched lines
2. Useful For
   * Transform data files
   * Produce formatted reports
3. Programming Constructs
   * Format output lines
   * Arithmetic and string operations
   * Conditionals and loops

# Syntax:

awk options 'script' input\_file > output\_file

awk options '/pattern/ action' input\_file > output\_file

Note : 'script' is in the form '/pattern/ action'

# Options

|  |  |
| --- | --- |
| Tag | Description |
| -F FS --field-separator FS | Use FS for the input field separator (the value of the 'FS' predefined variable). |
| -f PROGRAM-FILE  --file PROGRAM-FILE | Read the awk program source from the file PROGRAM-FILE, instead of from the first command line argument. |

# Built in variables in awk

awk’s built-in variables include the field variables—$1, $2, $3, and so on ($0 is the entire line) — that break a line of text into individual words or pieces called fields.

|  |  |
| --- | --- |
| Var. | Description |
| NR | NR command keeps a current count of the number of input records. Remember that records are usually lines. awk command performs the pattern/action statements once for each record in a file. |
| NF | NF command keeps a count of the number of fields within the current input record. |
| FS | FS command contains the field separator character which is used to divide fields on the input line. The default is “white space”, meaning space and tab characters. FS can be reassigned to another character (typically in BEGIN) to change the field separator. |
| RS | RS command stores the current record separator character. Since, by default, an input line is the input record, the default record separator character is a newline. |
| OFS | OFS command stores the output field separator, which separates the fields when awk prints them. The default is a blank space. |
| ORS | ORS command stores the output record separator, which separates the output lines when Awk prints them. The default is a newline character. |

**Testfile.txt**

Student Subject Marks Extra

stu1 sub1 m1 e1

stu2 sub2 m2 e2

stu2 sub3 m3

stu4 sub4 m4

# Useful Commands

## To print line having a pattern

$ awk '/sub2/ {print}' testfile.txt

stu2 sub2 m2 e2

## awk command with NR prints all the lines along with the line number

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

1 Student Subject Marks Extra

2 stu1 sub1 m1 e1

3 stu2 sub2 m2 e2

4

5 stu2 sub3 m3

6 stu4 sub4 m4

## Display first and last filed

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

Student Extra

stu1 e1

stu2 e2

stu2 m3

stu4 m4

## Display Line from 3 to 6

$ awk 'NR==3, NR==6 {print NR,$0}' testfile.txt

3 stu2 sub2 m2 e2

4

5 stu2 sub3 m3

6 stu4 sub4 m4

## To print any non-empty line if present

$ awk 'NF > 0' testfile.txt

Student Subject Marks Extra

stu1 sub1 m1 e1

stu2 sub2 m2 e2

stu2 sub3 m3

stu4 sub4 m4

## To find the length of the longest line present in the file

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

27

## To count the lines in a file

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

6

## Printing lines with more than 10 characters

$ awk 'length($0) > 10' testfile.txt

Student Subject Marks Extra

stu1 sub1 m1 e1

stu2 sub2 m2 e2

stu2 sub3 m3

stu4 sub4 m4

## To find/check for any string in any column

$ awk '{ if($3 == "m3") print $0;}' testfile.txt

stu2 sub3 m3

## To print the squares of first numbers from 1 to n

$ awk 'BEGIN { for(i=1;i<=5;i++) print "square of ", i, " is ",i\*i; }'

square of 1 is 1

square of 2 is 4

square of 3 is 9

square of 4 is 16

square of 5 is 25

## To count the lines in a file

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

6

# Summary of Commands

$ awk '/sub2/ {print}' testfile.txt

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

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

$ awk 'NR==3, NR==6 {print NR,$0}' testfile.txt

$ awk 'NF > 0' testfile.txt

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

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

$ awk 'length($0) > 10' testfile.txt

$ awk '{ if($3 == "m3") print $0;}' testfile.txt

$ awk 'BEGIN { for(i=1;i<=5;i++) print "square of ", i, " is ",i\*i; }'

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