|  |  |  |  |
| --- | --- | --- | --- |
|  | **what to do** | **awk command** | **meaning** |
| 1 | **start awk** | awk ‘{< awk commands>}’ | I am starting awk. The code in ‘ ‘ is all awk code |
| 2 | **print stuff**  \* print “hi” with awk  \* print a variable with awk  \* print given file with awk  \* do more than one thing inside awk | \* echo| awk '{print "hi"}' \* echo | awk '{a = "hi"; print a}'  \* echo | awk ‘{a = “hi”}{print a}’  \* cat <filename> | awk ‘{print $0}’ | \* declare “hi” as a variable  \* print the variable  \* if you try awk ‘{print “a b c”}’ without echo, **it will not work**  **\*** $0 means the entire line  \* Either separate every component with {} or put ; |
| 3 | **print file stuff**  \* print multiple lines  \* print nth token of each line / from the given file | \* echo -e "a b c \n d e f" | awk ‘{print $0}’  \* echo -e "<lines>”| awk '{print $(number n)}'  Eg: echo -e "a b c \n d e f"| awk '{print $3}'  cat <filename> | awk '{print $(number n)}' | echo -e means print each line. Each line is separated by ‘\n’  outputs  a b c (first eg)  d e f (first eg)  c (second eg)  f (second eg) |
| **4** | **how to know what each error means** | If you did something like  echo -e "a b c \n d e f"| awk '{print $4}'  Error  awk: 1: unexpected character 0xe2  awk: line <some line>: missing } near the end of file | Google it, the error. Read up first few things :  since there are not 4 tokens, you will get this error. |
| 5 | **replace** a character from each line | ….| awk '{gsub("character to replace", "character to replace it with ",token #); print}  Eg:  echo -e "a/b/c \nd/e/f"| awk '{gsub("/", "-",$0); print}' | prints  a-b-c  d-e-f  original text was  a/b/c  d/e/f  We replaced each line -- $0 , thing to replace, “/”, to replace it with, “-”. |
| 6 | **split** each line by given delimiter | ….| awk ‘{split ($0, <array saving the split>, “delimiter”; print array[1],...}’  Eg: echo -e "a/b/c \nd/e/f"| awk '{split($0, d, "/"); print d[1],d[2],d[3]}' | \* $0 for the entire line  \* ‘d’ is the name of array that saves the split.  **\* {print d} will not work. you need to print element by element. And elements start from index 1.** |
| 7 | \* get a **substring** from char\_start to char\_end  **note:** if char\_start == char\_end, you will get only THAT character  \* **save a substring as a variable** and print later | \* substr ($0, char\_start, char\_end)  eg:  echo -e "-1/2/0 \n4/5/6" | awk '{print substr($0,1,2)}'  \* {d = substr ($0, char\_start, char\_end)}{print d)  eg:  echo -e "-1/2/0 \n4/5/6" | awk '{d= substr($0,1,2);print d}' | \* $0 for each line. Start from 1st character, Include second character and end.  \* The substr is saved in a variable d  output  -1  4/ |
| 8 | **string concatenation** | {str1 =”...; str2=”...”, str3 = str1 str2; print str3}  Eg:  echo "Dhara" | awk '{str1 = substr($0,1,1); str2 = "Initial:" }{ print str2 str1}' | prints  Initial:D |
| 9 | **if statement** | if ( <if condition>) { do if thigns } else {do else things}  Eg:  echo -e "-3\n4"| awk '{  sign = substr($0,1,1);  if (sign == "-")  {print $0 " is a negative number"}  else  {print $0 " is a positive number"}  }' | -3is a negative number  4is a positive number |
| 10 | **run an awk file in a pipe** | ….| ./awkfile | you are saying that you need to run this awk file as a bash file on the output of the file |
| 11 | **Mathematical operatons**  \* division  \* reminder  \* convert a float to an integer | \*a/b Eg: $0/100  \*a%b Eg a % 100  \* int (a/b) | \* division gives a float  \* reminder gives an int  \* casts float to int |
| 12 | \* **code block** in awk  \* Perform a **column operation** | BEGIN {do the column operation} END {do something to the result}  Eg:  …..| awk ‘{sum = 0} BEGIN {sum += $0} END {print sum}’ | \* Declare variable sum = 0 (this is optional!)  \* sum up a file of integers (each line is an integer), and save them into sum  \* print that sum |

**Gallons, sickles, and knuts:**

* 1 gallon = 23 sickles
* so, sickles can range from 0...22. Sickles cannot have the value 23
* 1 sickle = 17 knuts
* Knuts can range from 0...16
* 1 gallon = 23\*17 knuts

**Converting Gallons, sickles, and knuts to knuts**

* x gallons/ y sickles / z knuts = x\*23\*17 knuts + y\*17 knuts + z knuts = (x\*23 + y)\*17 + z knuts
* if -x gallons/ y sickles / z knuts : take - sign apart, and only consider x/y/z → Convert to knuts → put the sign back

**Algorithm 1: convert s = “x/y/z” to w knuts:**

sign = “” // empty string

if (the first character of s == “-”)

{sign = “-”}

replace “-” in s with “” // you are converting the gallons to positive

w = (x\*23 + y)\*17 + z // put in the conversion formula

if (sign == “-”)

w = -w

* Eg: -10/7/9 → sign = “-”, convert 10/7/9 to knuts → sign = “-”, (10\*23 + 7)\*17 + 9 = (237\*17) + 9 = 4029 + 9 = 4038 knuts → put the sign back: -4038 knuts.

**Algorithm 2: Convert w knuts to G/S/K**

sign = “”

if (the first character of w == “-”)

{sign = “-”}

G = int ( w / (23\*17))

rem = w % (23\*17)

S = int (rem / 17)

K = rem % 17

ans = sign G “/” S “/” K // concatenate multiple strings

**Algorithm 3: sum of strings**

Input: file of strings

G1/S1/K1

G2/S2/K2

G3/S3/K3

……………..

output: string that represents sum of all these strings

Gt/St/Kt

Algorithm:

Step 1:

Convert input file to file of knuts with Algorithm 1

w1

w2

w3

……….

Step 2:

sum up all these integers (look at the last entry of the cheat sheet) to total knuts

wt

Step 3: convert wt knuts to

Gt/St/Kt

string using algorithm 2