gawk

MengChunlei

March 27, 2021

gawk 是一个文本处理工具。它用起来更加接近高级编程语言。它的基本格式为: gawk script file. 其中 script 是一段脚本程序，要放在单引号中。本文将按照以下 7 个部分来总结它的用法:

- 基本用法
- 变量
- 数组
- 模式
- 控制逻辑
- 格式化打印
- 函数

1 基本用法

1.1 提取字段

- $0: 整个文本
- $1: 本文中第一个字段
- $n: 文本中第 n 个字段

Listing 1: 1.1

```
mcl@mcl$ cat file1
data11 data12 data13
data21 data22 data23
data31 data32 data33
mcl@mcl$ gawk '{print $1, $3}' file1 /*默认用空格分割*/
data11 data13
data21 data23
data31 data33
mcl@mcl$ gawk '{print "The data is: " $1, $3}' file1
The data is: data11 data13
The data is: data21 data23
The data is: data31 data33
```

1.2 替换字段

Listing 2: 1.2

```
mcl@mcl$ gawk '{$3="hello"; print "The data is: " $1, $3}' file1
The data is: data11 hello
The data is: data21 hello
The data is: data31 hello
mcl@mcl$ echo "My name is alice" | gawk '{$4="mcl"; print $0}'
My name is mcl
mcl@mcl$ echo "My name is alice" | gawk '{  /*多个命令可以直接换行*/
> $4="mcl"
> print $0}'
My name is mcl
```

1.3 从文件读取命令

Listing 3: 1.3

```
mcl@mcl$ cat script1.gawk   /*将脚本存储在文件。同时可以看到可以用变量text*/
{
text = "'s home is: "
print $1 text $6
}
mcl@mcl$ gawk -F: -f script1.gawk /etc/passwd  /*-F可以设置分隔符*/
root's home is: /root
daemon's home is: /usr/sbin
bin's home is: /bin
sys's home is: /dev
sync's home is: /bin
games's home is: /usr/games
```

1.4 数据处理前后运行脚本

Listing 4: 1.4

```
mcl@mcl$ cat script2.gawk
BEGIN {                    /*BEGIN定义了数据处理前运行的一段脚本，只运行一次*/
print "The passwd info:"
FS=":"                              /*注意这里可以用FS设置分隔符*/
}
{
text = "'s home is: "
print $1 text $6
}
END{print "Data process finished"}    /*END定义了处理完后运行的脚本*/
mcl@mcl$ gawk -f script2.gawk /etc/passwd
The passwd info:
root's home is: /root
daemon's home is: /usr/sbin
bin's home is: /bin
Data process finished
```

2 变量

2.1 输入输出字段分隔符

Listing 5: 2.1

```
mcl@mcl$ cat file2
data11,data12,data13
data21,data22,data23
data31,data32,data33
mcl@mcl$ gawk 'BEGIN{FS=","} {print $1, $2, $3}' file2
data11 data12 data13
data21 data22 data23
data31 data32 data33
mcl@mcl$ gawk 'BEGIN{FS=","; OFS="-"} {print $1, $2, $3}' file2
data11-data12-data13  /*OFS可以设置输出的分隔符，默认是空格*/
data21-data22-data23
data31-data32-data33
```

2.2 按照宽度分割

Listing 6: 2.2

```
mcl@mcl$ cat file3
1234557890
3234557867
2929817231
mcl@mcl$ gawk 'BEGIN{FIELDWIDTHS="3 4 3"} {print $1, $2, $3}' file3
123 4557 890   /*FIELDWIDTHS决定了各个字段的宽度*/
323 4557 867
292 9817 231
```

2.3 输入输出记录分隔符

Listing 7: 2.3

```
mcl@mcl$ cat file4
xiaoming
qinghua
18810101010

xiaobai
renda
18812345678
mcl@mcl$ gawk 'BEGIN{FS="\n"; RS=""} {print $1, $3}' file4  /*换行为字段分隔符*/
xiaoming 18810101010                                        /*空行为记录分隔符*/
xiaobai 18812345678                                  /*默认输出记录分隔符为换行*/
mcl@mcl$ gawk 'BEGIN{FS="\n"; RS=""; ORS="\n==========\n"} {print $1, $3}' file4
xiaoming 18810101010          /*设定输出记录分隔符*/
==========
xiaobai 18812345678
==========
```

2.4 其他内置变量

Listing 8: 2. 4

```
mcl@mcl$ gawk 'BEGIN{print ARGC, ARGV[0], ARGV[1]}' file4     /*打印参数个数和参数*/
2 gawk file4
mcl@mcl$ gawk 'BEGIN{print ENVIRON["PATH"]}' file4        /*打印环境变量*/
/opt/ros/melodic/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin
mcl@mcl$ cat file2
data11,data12,data13
data21,data22,data23
data31,data32,data33
mcl@mcl$ gawk 'BEGIN{FS=","} {print $NF}' file2   /*NF代表了总的字段个数*/
data13
data23
data33
mcl@mcl$ gawk '
BEGIN {FS=""}
{print $1, "FNR="FNR, "NR="NR}
END{print "The total records: "NR}' file2 file2   /*注意这里文件被处理了两次*/
d FNR=1 NR=1          /*FNR表示当前文件已经处理的记录个数*/
d FNR=2 NR=2          /*NR表示当前命令已经处理的记录个数*/
d FNR=3 NR=3
d FNR=1 NR=4
d FNR=2 NR=5
d FNR=3 NR=6
The total records: 6
```

2.5 自定义变量

Listing 9: 2.5

```
mcl@mcl$ gawk 'BEGIN {    /*设置变量; 变量可以被改变;    可以进行数学运算*/
text="this is test"; print text
text=1209; test=text*2
print text}'
this is test
1209
mcl@mcl$ cat script3.gawk
BEGIN{FS=","; print "the n=", n}
{print $n}
mcl@mcl$ gawk -f script3.gawk n=3 file2   /*设置的n=3对  BEGIN无效*/
the n=
data13
data23
data33
mcl@mcl$ gawk -v n=3 -f script3.gawk file2   /*BEGIN中的变量需要用-v设置*/
the n= 3
data13
data23
data33
```

3 数组

3.1 数组定义

Listing 10: 3.1

```
mcl@mcl$ gawk 'BEGIN{
mapper["data11"]=100
mapper["data21"]=2000
mapper["data31"]=30000
}
{print mapper[$1]}' file1
100
2000
30000
```

3.2 数组遍历和删除

Listing 11: 3.2

```
mcl@mcl$ gawk 'BEGIN{
mapper["data11"]=100
mapper["data12"]=2000
mapper["data31"]=30000
for (e in mapper) {
  print "key: ", e, " -> ", mapper[e]
}
}'
key:  data11  ->  100
key:  data12  ->  2000
key:  data31  ->  30000
mcl@mcl$ gawk 'BEGIN{
mapper["data11"]=100
mapper["data12"]=2000
mapper["data31"]=30000
for (e in mapper) {
  print "key: ", e, " -> ", mapper[e]
}
delete mapper["data12"]
print "====="
for (e in mapper) {
  print "key: ", e, " -> ", mapper[e]
}
}'
key:  data11  ->  100
key:  data12  ->  2000
key:  data31  ->  30000
=====
key:  data11  ->  100
key:  data31  ->  30000
```

4 模式

4.1 正则表达式

Listing 12: 4.1

```
mcl@mcl$ cat file2
data11,data12,data13
data21,data22,data23
data31,data32,data33
mcl@mcl$ gawk '/22/{print $0}' file2   /*包含22的行会被处理*/
data21,data22,data23
```

4.2 匹配符

Listing 13: 4.2

```
mcl@mcl$ gawk 'BEGIN{FS=","} $2 ~ /data[2,3]/{print $0}' file2
data21,data22,data23                    /*第二个字段包含data2或者data3*/
data31,data32,data33
mcl@mcl$
mcl@mcl$ gawk 'BEGIN{FS=","} $2 !~ /data[2,3]/{print $0}' file2
data11,data12,data13                    /*第二个字段不包含data2或者data3*/
```

4.3 数学表达式

Listing 14: 4.3

```
mcl@mcl$ cat file3
1234557890
3234557867
2929817231
mcl@mcl$ gawk 'BEGIN{FIELDWIDTHS="3 4 3"} $1 < 300{print $1, $2, $3}' file3
123 4557 890
292 9817 231
```

5 控制逻辑

5.1 if

Listing 15: 5.1

```
mcl@mcl$ cat file5
23
19
15
mcl@mcl$ cat script4.gawk
{
if ($1 > 20) {
  print $1 * 2
} else {
  print $1 + 10
}
```

```
12 }
13 mcl@mcl$ gawk -f script4.gawk file5
14 46
15 29
16 25
```

5.2 while

Listing 16: 5.2

```
1  mcl@mcl$ cat file6
2  12 3 10 5
3  9 12
4  4 6 30
5  mcl@mcl$ cat script5.gawk
6  {
7  total = 0
8  i = 1
9  while (i <= NF) {
10   total += $i
11   ++i
12 }
13 avg = total / NF
14 print "average is:", total, "/", NF, "=", avg
15 }
16 mcl@mcl$ gawk -f script5.gawk file6
17 average is: 30 / 4 = 7.5
18 average is: 21 / 2 = 10.5
19 average is: 40 / 3 = 13.3333
```

5.3 for

Listing 17: 5.3

```
1  mcl@mcl$ cat script6.gawk
2  {
3  total = 0
4  for (i = 1; i <= NF; ++i) {
5    total += $i
6  }
7  avg = total / NF
8  print "average is:", total, "/", NF, "=", avg
9  }
10 mcl@mcl$ gawk -f script6.gawk file6
11 average is: 30 / 4 = 7.5
12 average is: 21 / 2 = 10.5
13 average is: 40 / 3 = 13.3333
```

6 格式化打印

6.1 各种类型

Listing 18: 6.1

```
mcl@mcl$ cat script7.gawk
BEGIN{
x = 75
printf "  ascii: %c\n", x
printf "integer: %d\n", x
printf "      e: %e\n", x
printf "  float: %f\n", x
printf "  octal: %o\n", x
printf "    hex: %x\n", x
printf "    HEX: %X\n", x
}
mcl@mcl$ gawk -f script7.gawk
     ascii: K
integer: 75
       e: 7.500000e+01
   float: 75.000000
   octal: 113
     hex: 4b
     HEX: 4B
```

6.2 对齐和宽度

Listing 19: 6.2

```
mcl@mcl$ cat file7
zhangsan 19801212 93.45
lisi 19900916 88.5
wangwu 20050304 70.123
mcl@mcl$ gawk '{printf "%9s %s %.5f\n", $1, $2, $3}' file7
 zhangsan 19801212 93.45000     /*靠右对齐*/
     lisi 19900916 88.50000
   wangwu 20050304 70.12300
mcl@mcl$ gawk '{printf "%-9s %s %.5f\n", $1, $2, $3}' file7
zhangsan  19801212 93.45000    /*靠左对齐*/
lisi      19900916 88.50000
wangwu    20050304 70.12300
```

7 函数

7.1 数学函数

Listing 20: 7.1

```
mcl@mcl$ cat script8.gawk
BEGIN{
pi = 3.1415926
printf "    sin(30): %.4f\n", sin(pi / 6)
printf "    cos(30): %.4f\n", cos(pi / 6)
printf "     exp(2): %.4f\n", exp(2)
printf "    int(pi): %.4f\n", int(pi)
printf "log(exp(2)): %.4f\n", log(exp(2))
```

```
9  printf "      rand(): %.4f\n", rand()   /*rand返回的是[0,1]之间的小数*/
10 printf "      rand(): %.4f\n", rand()
11 printf "     sqrt(9): %.4f\n", sqrt(9)
12 printf "   and(5,6): %d\n", and(5, 6)
13 printf "    or(5,6): %d\n", or(5, 6)
14 printf "   xor(5,6): %d\n", xor(5, 6)
15 printf "lshift(5,2): %d\n", lshift(5,2)
16 printf "rshift(5,2): %d\n", rshift(5,2)
17 }
18 mcl@mcl$ gawk -f script8.gawk
19      sin(30): 0.5000
20      cos(30): 0.8660
21       exp(2): 7.3891
22      int(pi): 3.0000
23 log(exp(2)): 2.0000
24       rand(): 0.2378
25       rand(): 0.2911
26      sqrt(9): 3.0000
27     and(5,6): 4
28      or(5,6): 7
29     xor(5,6): 3
30 lshift(5,2): 20
31 rshift(5,2): 1
```

7.2 字符串函数

Listing 21: 7.2

```
1  mcl@mcl$ cat script9.gawk
2  BEGIN{
3  s="11, abc, 11, pqr"
4  printf "          gensub: %s\n", gensub("11", "22", 1, s) /*替换第一次11*/
5  printf "          gensub: %s\n", gensub("11", "22", 2, s)
6  printf "          gensub: %s\n", gensub("11", "22", "g", s) /*替换所有11*/
7  printf "          gensub: %s\n", gensub("a[a-z]{2}", "xxx", "g", s) /*支持正则表达式*/
8  printf "gsub-replace-num: %d\n", gsub("11", "22", s) /*将所有11替换，返回替换的个数*/
9  printf " replace-by-gsub: %s\n", s  /*s被改变*/
10 s="11, abc, 11, pqr"
11 printf "           index: %d\n", index(s, "abc") /*返回第一次出现的位置,下标从1开始*/
12 printf "           index: %d\n", index(s, "xxx") /*没有找到返回0*/
13 printf "          length: %d\n", length(s)
14 printf "           match: %d\n", match("xx11,11", "11") /*返回第一次match的起始位置*/
15 printf "           match: %d\n", match("xx11,11", "11", a)
16 for (x in a) {  /*数组a中保存了第一次匹配的位置和长度*/
17 print x, " -> ", a[x]
18 }
19 printf "           split: %d\n", split("xx11,11,1323", b, ",") /*将逗号分隔的结果放到b*/
20 for (x in b) {
21 print x, " -> ", b[x]
22 }
23 printf "         sprintf: %s\n", sprintf("format data: [%s], %d", s, 123)
24 printf "          substr: %s\n", substr(s, 5, 3)
25 printf "         toupper: %s\n", toupper("abc")
```

```
26 printf "          tolower: %s\n", tolower("abcDEF")
27 }
```

```
mcl@mcl$ gawk -f script9.gawk
          gensub: 22, abc, 11, pqr
          gensub: 11, abc, 22, pqr
          gensub: 22, abc, 22, pqr
          gensub: 11, xxx, 11, pqr
gsub-replace-num: 2
 replace-by-gsub: 22, abc, 22, pqr
           index: 5
           index: 0
          length: 16
           match: 3
           match: 3
0start  ->  3
0length  ->  2
0  ->  11
           split: 3
1  ->  xx11
2  ->  11
3  ->  1323
         sprintf: format data: [11, abc, 11, pqr], 123
          substr: abc
         toupper: ABC
         tolower: abcdef
```

7.3 时间函数

Listing 22: 7.3

```
1  mcl@mcl$ cat script10.gawk
2  BEGIN{
3  date = systime()
4  day = strftime("%A, %b %d, %Y", date)
5  print day
6  tp = mktime("2021 03 27 12 58 30")
7  print tp
8  print strftime("%A, %b %d, %Y", tp)
9  }
10 mcl@mcl$ gawk -f script10.gawk
11 星期六, 3月 27, 2021
12 1616821110
13 星期六, 3月 27, 2021
```

7.4 自定义函数

Listing 23: 7.4

```
1  mcl@mcl$ cat  script11.gawk
2  function myprint(a, b, c) {
3    printf "%10s %d %.5f\n", a, b, c
4  }
```

```
{
myprint($1, $2, $3)
}
mcl@mcl$ gawk -f script11.gawk file7
zhangsan 19801212 93.45000
    lisi 19900916 88.50000
  wangwu 20050304 70.12300
```

7.5 函数库

Listing 24: 7.5

```
mcl@mcl$ cat funclib.gawk
function print2(a, b) {
  printf "%10s %d", a, b
}

function print_third() {
  printf " %.4f\n", $3
}

mcl@mcl$ cat script12.gawk
{
print2($1, $2)
print_third()
}
mcl@mcl$ gawk -f funclib.gawk -f script12.gawk file7
zhangsan 19801212 93.4500
    lisi 19900916 88.5000
  wangwu 20050304 70.1230
```