1. 操作符优先级

Operators

The operators in AWK, in order of decreasing precedence, are

Awk中的操作符优先级以递减顺序如下排列

(...) Grouping

$ Field reference.

++ -- Increment and decrement, both prefix and postfix.

^ Exponentiation (\*\* may also be used, and \*\*= for the assignment operator).

+ - ! Unary plus, unary minus, and logical negation.

\* / % Multiplication, division, and modulus.

+ - Addition and subtraction.

space String concatenation.

| |& Piped I/O for getline, print, and printf.

< >

<= >=

!= == The regular relational operators.

~ !~ Regular expression match, negated match. NOTE: Do not use a constant regular expression (/foo/) on the left-hand side of a ~ or !~. Only use one on the right-hand side. The expression /foo/ ~ exp has the same meaning as (($0 ~ /foo/) ~ exp). This is usually not what was intended.

in Array membership.

&& Logical AND.

|| Logical OR.

?: The C conditional expression. This has the form expr1 ? expr2 : expr3. If expr1 is true, the value of the expression is expr2, otherwise it is expr3. Only one of expr2 and expr3 is evaluated.

= += -=

\*= /= %= ^=

Assignment. Both absolute assignment (var = value) and operator-assignment (the other forms) are sup-ported.

正则匹配表示方法

$0~"" 字符串动态正则（字符串计算）

$0~x 变量动态正则（变量计算）

$0~// 常量(静态)正则

1. 例题
2. seq 10 23 | awk -vx="1" '$0~1x'

'$0~1x' ==> '$0~(1x)' ==> '$0~(11)' ==> '$0~11'

先用数值常量1和变量x做连接操作,若变量x为空则匹配1,否则匹配连接后的字符串或数字

1. seq 10 23 | awk -vx="1" '$0~1 x'

'$0~1 x' ==> '$0~(1 x)' ==> '$0~11'

空格优先级高于~,因此先用数值常量1和变量x连接操作,然后匹配

1. seq 10 23 | awk -vx="1" '$0~x3'

'$0~x3' ==> '$0~(x3)' ==> '$0~(空)' ==> '$0~""'

x3被识别为变量,而这里定义的变量x值为1,变量x3没有定义,其值默认为null,而每个record的各个字符之间均为null,所以所有行都会被匹配,全部打印出来

1. seq 10 23 | awk -vx="1" '$0~x 3'

'$0~x 3' ==> '$0~(x 3)' ==> '$0~(13)' ==> '$0~13'

空格优先级高于~,因此先用变量x和数值常量3连接操作,然后匹配

1. seq 10 23 | awk -vx="3" '$0~/1/x'

'$0~/1/x' ==> '$0~((/1/)x)'

先使用常量正则匹配数值1,

匹配到1为真,返回布尔值1,和变量x连接后动态正则匹配$0~13

未匹配到1为假,返回布尔值0,和变量x连接后为动态正则匹配$0~03

1. seq 10 23 | awk -vx="" '$0~/1/x 3''

'$0~/1/x 3' ==> '$0~(/1/(x 3))' ==> '$0~(/1/(3))'

使用常量正则匹配数值1, 用变量x和数值常量3连接操作

匹配到1为真,返回布尔值1,和数值常量3连接后动态正则匹配$0~13

未匹配到1为假,返回布尔值0,和数值常量3连接后为动态正则匹配$0~03

1. echo -e '1x11x\n22xx\n11xx\n111xx\n111xxxx ' | awk '$0~/1+xx/'

匹配一个或多个字符串常量1,后面跟了字符串xx的行

1. echo -e '1x11x\n22xx\n11xx\n111xx\n111xxxx ' | awk -vxx=1 '$0~1+"xx"'

'$0~1+"xx"' ==> '$0~(1+0)' ==> '$0~1'

+优先级高于~,先用数值常量1和字符串xx做加法运算,字符串在数值运算中的值为0,和1相加后结果为1,然后动态匹配$0~1

双引号中的xx为普通字符串而不是变量

1. echo -e '1x11x\n22xx\n11xx\n111xx\n111xxxx ' | awk '$0~1+/xx/'

awk '$0~1+/xx/' ==> '$0~(1+($0~/xx/))'

先使用常量正则匹配数值字符串xx,

匹配到xx为真,返回布尔值1,和数值常量1做加法数值运算后动态正则匹配$0~2

未匹配到xx为假,返回布尔值0,和数值常量1做加法数值运算后动态正则匹配$0~1