正则表达式与文件格式化处理

基础正则表达式

特殊符号	代表意义
[:alnum:]	代表英文大小写字符及数字，亦即 0-9, A-Z, a-z
[:alpha:]	代表任何英文大小写字符，亦即 A-Z, a-z
[:blank:]	代表空白键与 [Tab] 按键两者
[:cntrl:]	代表键盘上面的控制按键，亦即包括 CR, LF, Tab, Del.. 等等
[:digit:]	代表数字而已，亦即 0-9
[:graph:]	除了空白字符 （空白键与 [Tab] 按键） 外的其他所有按键
[:lower:]	代表小写字符，亦即 a-z
[:print:]	代表任何可以被打印出来的字符
[:punct:]	代表标点符号 （punctuation symbol），亦即：" ' ? ! ; : # $...
[:upper:]	代表大写字符，亦即 A-Z
[:space:]	任何会产生空白的字符，包括空白键, [Tab], CR 等等
[:xdigit:]	代表 16 进位的数字类型，因此包括： 0-9, A-F, a-f 的数字与字符

- 特殊符号及其代表意义
  - 不同语系编码不同导致输出结果可能不同 — 原因 — 语系影响
- grep [-A] [-B] [--color=auto] '搜寻字符串' filename — 用法
  - grep高级参数
    - 后接数字；为after之意，除列出本行外后续n行一并列出 — -A
    - 后接数字；为before之意，除列出本行外前面n行也列出 — -B
    - 选项与参数
    - 将正确的选取数据上色（CentOS7中默认alias） — --color=auto

RE 字符	意义与范例
^word	意义：待搜寻的字串(word)在行首！ 范例：搜寻行首为 # 开始的那一行，并列出行号 **grep -n '^#' regular_express.txt**
word$	意义：待搜寻的字串(word)在行尾！ 范例：将行尾为 ! 的那一行列印出来，并列出行号 **grep -n '!$' regular_express.txt**
.	意义：代表『一定有一个任意字节』的字符。 范例：搜寻的字串可以是 (eve) (eae) (eee) (e e)， 但不能仅有 (ee)！亦即 e 与 e 中间『一定』仅有一个字节，而空白字节也是字节！ **grep -n 'e.e' regular_express.txt**
\	意义：跳脱字符，将特殊符号的特殊意义去除！ 范例：搜寻含有单引号 ' 的那一行！ **grep -n '\' regular_express.txt**
*	意义：重复零个到无穷多个的前一个 RE 字符 范例：找出含有 (es) (ess) (esss) 等等的字串，注意，因为 * 可以是 0 个，所以 es 也是符合带搜寻字串。另外，因为 * 为重复『前一个 RE 字符』的符号， 因此，在 * 之前必须要紧接着一个 RE 字符喔！例如任意字节则为『.*』！ **grep -n 'ess*' regular_express.txt**
[list]	意义：字节集合的 RE 字符，里面列出想要撷取的字节！ 范例：搜寻含有 (gl) 或 (gd) 的那一行，需要特别留意的是，在 [] 当中『谨代表一个待搜寻的字节』，例如『a[df]y』代表搜寻的字串可以是 aay, afy, aly 即 [df] 代表 a 或 f 或 l 的意思！ **grep -n 'g[ld]' regular_express.txt**
[n1-n2]	意义：字节集合的 RE 字符，里面列出想要撷取的字节范围！ 范例：搜寻含有任意数字的那一行！需特别留意，在字节集合 [] 中的减号 - 是有特殊意义的，他代表两个字节之间的所有连续字节！但这个连续与否与 ASCII 编码有关，因此，你的编码需要配置正确(在 bash 当中，需要确定 LANG 与 LANGUAGE 的变量是否正确！) 例如所有大写字节则为 [A-Z] **grep -n '[A-Z]' regular_express.txt**
[^list]	意义：字节集合的 RE 字符，里面列出不要的字串或范围！ 范例：搜寻的字串可以是 (oog) (ood) 但不能是 (oot)， 那个 ^ 在 [] 内时，代表的意义是『反向选择』的意思。 例如，我不要大写字节，则为 [^A-Z]。但是，需要特别注意的是，如果以 grep -n [^A-Z] regular_express.txt 来搜寻，却发现该文件内的所有行都被列出，为什么？因为这个 [^A-Z] 是『非大写字节』的意思， 因为每一行均有非大写字节，例如第一行的 "Open Source" 就有 p,e,n,o,... 等等的小写字 **grep -n 'oo[^t]' regular_express.txt**
\{n,m\}	意义：连续 n 到 m 个的『前一个 RE 字符』 意义：若为 \{n,\} 则是连续 n 个的前一个 RE 字符。 意义：若是 \{n,\} 则是连续 n 个以上的前一个 RE 字符。范例：在 g 与 g 之间有 2 个到 3 个的 o 存在的字串，亦即 (goog)(gooog) **grep -n 'go\{2,3\}g' regular_express.txt**

- 特殊字符

扩展正则表达式

RE 字符	意义与范例
+	意义：重复"一个或一个以上"的前一个 RE 字符 范例：搜寻 (god) (good) (goood) ... 等等的字串。 那个 o+ 代表"一个以上的 o "所以，下面的执行结果会将第 1, 9, 13 行列出来。 **egrep -n 'go+d' regular_express.txt**
?	意义："零个或一个"的前一个 RE 字符 范例：搜寻 (gd) (god) 这两个字串。 那个 o? 代表"空的或 1 个 o "所以，上面的执行结果会将第 13, 14 行列出来。有没有发现到，这两个案例（'go+d' 与 'go?d'）的结果集合与 'go*d' 相同？ 想想看，这是为什么喔！ ^_^ **egrep -n go?d' regular_express.txt**
|	意义：用或（or）的方式找出数个字串 范例：搜寻 gd 或 good 这两个字串，注意，是"或"！ 所以，第 1,9,14 这三行都可以被打印出来喔！ 那如果还想要找出 dog 呢？ **egrep -n 'gd|good' regular_express.txt** **egrep -n 'gd|good|dog' regular_express.txt**
()	意义：找出"群组"字串 范例：搜寻 (glad) 或 (good) 这两个字串，因为 g 与 d 是重复的，所以， 我就可以将 la 与 oo 列于 ( ) 当中，并以 | 来分隔开来，就可以做！ **egrep -n 'g(la|oo)d' regular_express.txt**
() +	意义：多个重复群组的判别 范例：将"AxyzxyzxyzxyzxyzC"用 echo 叫出，然后再使用如下的方法搜寻一下！ **echo 'AxyzxyzxyzxyzC' | egrep 'A(xyz)+C'** 上面的例子意思是说，我要找开头是 A 结尾是 C ，中间有一个以上的 "xyz" 字串的意思～

- 特殊RE字符

文件格式化及相关处理

- printf
  - 功能 — 格式化打印数据
  - 用法 — printf '打印格式' 实际内容
  - 选项与参数 — 特殊样式
    - \a — 警告声音 输出
    - \b — 倒退键（backspace）
    - \f — 清除屏幕（form feed）
    - \n — 输出新的一行
    - \r — 等于Enter键
    - \t — 水平的tab键
    - \v — 垂直的tab键
    - \xNN — NN为两位数字，可转换数字为字符
- awk
  - 功能 — 数据处理工具，倾向于将一行分成数个"字段"来处理（以行为一次处理的单位，以字段为最小的处理单位）
  - 用法 — awk '条件类型1 {动作1} 条件类型2 {动作2}.....' 条件类型 n {动作n}' filename

变量名称	代表意义
NF	每一行 （so） 拥有的字段总数
NR	目前 awk 所处理的是"第几行"数据
FS	目前的分隔字符，默认是空白键

  - 内置变量及其含义

运算单元	代表意义
>	大于
<	小于
>=	大于或等于
<=	小于或等于
==	等于
!=	不等于

  - 条件类型
    - 逻辑运算符及其含义
    - 关键字
      - BEGIN — 预先设置内置变量
      - END
  - 特性
    - 指令间隔 — 所有awk的动作，即()内的动作：1.可利用分号";"间隔 2.直接以Enter键隔开每个指令
    - 逻辑运算 — 等于务必使用"=="
    - 格式化输出 — printf格式设置中，务必加上\n才能分行
    - 变量取用 — 不同于bash，awk中变量可直接取用不需加上$符号
- 文件比较工具
  - diff
    - 功能 — 比对两文件之间的差异，且以"行"为单位来对比（常用来比较同一文件的新旧版本）对比两目录之间的差异
    - 用法 — diff [-bBi] from-file to-file — from-file：文件名，作为原始比较文件的文件名 to-file：文件名，作为目的比较文件的文件名（以上二者可由-取代，代表stdin）
    - 选项与参数
      - -b — 忽略一行中仅存在多个空白的差异（a b和a b相同）
      - -B — 忽略空白行的差异
      - -i — 忽略大小写的不同
  - cmp
    - 功能 — 比对两文件之间的差异，且以"字节"为单位来对比（可以比较二进制文件）
    - 用法 — cmp [-l] file1 file2
    - 选项与参数 — -l — 将所有的不同点得字节处都列出（默认仅会输出第一个发现的不同点）
  - patch
    - 功能 — 将旧文件接补丁文件升级成新文件
    - 用法
      - 更新 — patch -pN < patch_file
      - 还原 — patch -R -pN < patch_file
    - 选项与参数
      - -p — 后接的N代表取消几层目录的意思
      - -R — 代表还原
- 文件打印准备 — pr — 设置页码、标题等以打印文件

试用版 XMind:ZEN 试用版