grep 快速参考

模式匹配、正则、递归搜索、上下文、过滤

基础用法

运行 grep

```
grep "pattern" file.txt            # search in file
grep "error" *.log                 # search multiple files
grep "hello" file1.txt file2.txt   # explicit file list
cat file.txt | grep "pattern"      # pipe input
dmesg | grep -i "usb"              # filter command output
```

常用选项

-i	不区分大小写
-v	反向匹配——打印不匹配的行
-c	打印匹配行数
-n	显示行号
-l	仅列出含匹配的文件名
-L	仅列出不含匹配的文件名
-w	仅匹配完整单词
-x	仅匹配整行

正则模式

基础正则（BRE）

.	任意单个字符
*****	前一元素零个或多个
^	行首
$	行尾
[abc]	字符类——a、b、c 之一
[^abc]	否定类——非 a、b、c
[a-z]	范围——小写字母
\<, \>	词边界（GNU）
\(\), \1	捕获组与反向引用

BRE 示例

```
grep '^#' file.conf                # lines starting with #
grep 'error$' file.log             # lines ending with error
grep '^$' file.txt                 # blank lines
grep 'col[ou]r' file.txt           # match color or colour
```

扩展正则

扩展正则（ERE）

+	前一元素一个或多个
?	前一元素零个或一个
{n}	精确重复 n 次
{n,m}	重复 n 到 m 次
(a|b)	交替——匹配 a 或 b
( )	分组（无需反斜杠）

ERE 示例

```
grep -E '[0-9]{3}-[0-9]{4}' f     # phone number pattern
grep -E '(error|warn|fatal)' f     # multiple patterns
grep -E '^[A-Z][a-z]+' f           # capitalized words
grep -P '\d{1,3}\.\d{1,3}' f      # Perl regex: IP fragments
```

上下文行

上下文示例

```
grep -B 3 "error" app.log          # 3 lines before match
grep -A 5 "FAIL" test.log          # 5 lines after match
grep -C 2 "crash" kern.log         # 2 lines before and after
grep --group-separator="---" -C 1 "err" f  # custom separator
```

上下文选项

-B N	显示匹配前 N 行
-A N	显示匹配后 N 行
-C N	显示匹配前后各 N 行
--group-separator=str	匹配组之间的分隔符（默认 **--**）
--color=auto	在终端高亮匹配内容

递归搜索

递归示例

```
grep -r "TODO" .                   # recursive from current dir
grep -rn "FIXME" src/              # recursive with line numbers
grep -r --include="*.py" "import" .  # only .py files
grep -r --exclude="*.log" "error" .  # skip .log files
grep -r --exclude-dir=node_modules "require" .
```

递归选项

-r / --recursive	递归搜索目录
-R	类似 **-r** 但跟随符号链接
--include=glob	仅搜索匹配 glob 的文件
--exclude=glob	跳过匹配 glob 的文件
--exclude-dir=dir	跳过匹配名称的目录
--include-dir=dir	仅搜索匹配名称的目录

计数与列出

计数与列出示例

```
grep -c "error" *.log              # count matches per file
grep -l "TODO" src/*.py            # list files with TODOs
grep -L "test" src/*.py            # files missing "test"
grep -o "http[^ ]*" page.html      # extract matching parts only
grep -c '' file.txt                # count total lines (like wc -l)
```

输出选项

-c	每个文件的匹配行数
-l	仅打印含匹配的文件名
-L	仅打印不含匹配的文件名
-o	仅打印匹配的部分
-H / -h	显示 / 隐藏文件名前缀
-Z	null 分隔输出（配合 xargs -0）

反向匹配

取反与排除

```
grep -v "^#" config.conf           # remove comment lines
grep -v "^$" file.txt              # remove blank lines
grep -v -e "debug" -e "trace" app.log  # exclude two patterns
grep -v "pattern" f | grep "other" # chain: NOT A, then B
```

过滤策略

-v	反向匹配——选出不匹配的行
-v with -e	排除多个模式
pipe chain	链式 grep 实现复杂过滤
grep -v '^$' | grep -v '^#'	去除空行和注释行
-v with -c	统计不匹配的行数

多模式

多模式示例

```
grep -e "error" -e "warning" app.log
grep -E "error|warning|fatal" app.log
grep -f patterns.txt file.txt      # patterns from file
grep -w -e "GET" -e "POST" access.log
```

模式选项

-e pattern	指定模式（可多次使用）
-f file	从文件读取模式（每行一个）
-E 'a|b|c'	ERE 交替匹配多个模式
-F	固定字符串——无正则，速度更快
-G	基础正则（默认模式）
-P	Perl 兼容正则（PCRE）

性能

性能技巧

-F (fgrep)	固定字符串模式——字面量搜索最快
LC_ALL=C grep	绕过 locale，ASCII 数据快 2-10 倍
--include/--exclude	打开文件前先缩小搜索范围
-m N	每个文件匹配 N 次后停止
-q	静默模式——第一次匹配即退出（脚本用）
ripgrep (rg)	兼容替代；大仓库速度更快

性能示例

```
LC_ALL=C grep -F "exact string" huge.log
grep -r -m 1 "needle" /var/log/     # stop after first hit
grep -rq "pattern" . && echo "found" # boolean test
grep -r --include="*.go" "func main" .
```

常用模式

一行命令

```
grep -rn "TODO\|FIXME\|HACK" src/      # find code markers
grep -oP '(?<=")[^"]+(?=")' f          # extract quoted strings
grep -E '^\s*$' f | wc -l              # count blank lines
grep -c '' *.py | sort -t: -k2 -rn     # sort files by line count
grep -rn --include="*.yaml" "password" . # audit for secrets
```

实用示例

IP 地址	**grep -oE '[0-9]{1,3}(\.[0-9]{1,3}){3}'**
邮箱地址	**grep -oE '[a-zA-Z0-9._%+-]+@[a-z.-]+'**
URL	**grep -oE 'https?://[^]+'**
标记间的行	**grep -A999 'START' f | grep -B999 'END'**
唯一匹配	**grep -oE 'pattern' f | sort -u**
按模式计数	**grep -c 'pat1' f; grep -c 'pat2' f**