3 Przeszukiwanie - wyrażenia regularne, grep

Wyszukiwanie tekstu - grep

Bardzo ważną operacją jest możliwość przeszukiwania zawartości plików (tekstowych). Sprowadza się to do wyszukania zadanego tekstu (lub wzorca) w konkretnym pliku bądź zbiorze plików. Taką funkcjonalność udostępnia komenda `grep`. Podstawowe użycie:

`grep 'slowo' tekst.in` (szuka tekstu *slowo* w pliku *tekst.in*)
`grep 'slowo' file1 file2 ...` (wyszukuje *slowo* w podanych plikach)
`grep 'slowo' < tekst.in` (wyszukuje *slowo* w tekscie ze standardowego wejscia)

Domyślnie `grep` wypisuje wszystkie linie zawierające wskazany wzorzec.

Możliwość wyszukiwania w standardowym wejściu oraz możliwość podania wielu plików jako argumenty, pozwala na wygodne łącznie innych komend poprzez `'|'` bądź `xargs`.

Parametry grepa

Sam `grep` może też wpłynąć na sposób działania wyszukiwania:

`grep -c`	wypisuje tylko liczbę linii zawierających wzorzec
`grep -i 'iGnORE'`	ignoruje wielkośc liter
`grep -n`	wypisuje znalezione linie wraz z numerami tych linii
`grep -v`	negacja: wypisuje linie w których *nie* występuje wzorzec
`grep -f wzorce.in tekst.in`	wyszukuje wszystkie wzorce z pliku „wzorzec.in" (jeden wzorzec musi zawierać się w jednej linii)
`grep -r katalog`	wyszukuje rekurencyjnie we wszystkich plikach we wszytkich podkatalogach (oprócz wystapienia wypisuje nazwę pliku z którego pochodzi dana linijka)
`grep -w`	ogranicza się do linii gdzie wzorzecz występuje jako całe słowo
`grep -o`	zamiast całej linii wypisuje tylko dopasowany fragment (ma znaczenie przy użyciu wzorców)
`grep -l`	wypisuje same nazwy plików, w których występuje wzorzec
	dla każdego wystąpienia (kontekst):
`grep -A 3`	- wypisz dodatkowo 3 linie po nim
`grep -B 3`	- wypisz dodatkowo 3 linie przed nim
`grep -C 3`	- wypisz dodatkowo 3 linie przed i po nim

Wyszukiwanie wzorca

Program `grep` umożliwia bardziej zaawansowane wyszukiwanie tekstu - zamiast konkretnego słowa, możemy podać wzorzec zawierający znaki specjalne. Do opisu tych wzorców wykorzystuje się *wyrażenia regularne*, których składnia zostanie podana poniżej. W tej sekcji podajemy kilka uwag dotyczących pracy grepa z wyrażeniami regularnymi.

Tryby pracy
Jest kilka składni wyrażeń regularnych - `grep` obsługuje trzy rodzaje. Wywołany z flagą:
`-G` interpretuje wzorzec jako basic regular expression (BRE), wersja domyślna;
`-E` interpretuje wzorzec jako extended regular expression (ERE);
`-P` interpretuje wzorzec jako perlowe wyrażenie regularne (PCRE).

Wyrażenia perlowe posiadają najwięcej różnych funkcji i są bardzo rozpowszechnione także w innych językach. Pozostałe rodzaje są mniej bogate, ale też są podobne. W poniższych przykładach, gdyby któraś konstrucja nie działała, proszę zastosować `grep -P`. Bez opcji `-P` trudniej podaje się znaki specjalne (np. `"a\|b"` zamiast samego `"a|b"`).

Znaki specjalne
Wyrażenia regularne, podobnie jak globy, korzystają ze znaków specjalnych - np. takich jak `*`. Ponieważ wszystkie te znaki specjalne mają być analizowane przez `grep` a nie przez wbudowany w basha glob, prawie zawsze umieszcza się wzorzec w pojedyncze apostrofy – tak aby bash go przypadkiem nie zmodyfikował.

Wyrażenia regularne

W poprzedniej części kursu poznaliśmy bardzo prosty system wyszukiwania wzorców w postaci globów. Jest on prosty w użyciu, ale zarazem bardzo ograniczony i skupiony głównie na przeszukiwaniu nazw plików. Wyrażenia regularne w ogólnym podejściu pozwalają na wyszukiwanie bardziej skomplikowanych wzorców w dowolnym tekście.

Wyrażenia regularne posiadają swoją składnię, opisaną poniżej. Do pracy z bardziej skomplikowanymi wyrażeniami pomocne może być narzędzie `https://regex101.com`.

Zbiory znaków

Jeśli na danej pozycji tekstu może pojawić się nie jedna konkretna litera ale któraś z większego zbioru, to taki zbiór określamy używając nawiasów kwadratowych, podobnie jak w globach:

`[0123456789]` linia zawierająca co najmniej jedną cyfrę
`[0-9]` to samo, co wyżej, ale krócej zapisane
`[A-Za-z0-9_]` linia z co najmniej jedną literą, cyfrą lub podkreślnikiem
`T[a-z][aeiou]` fragment zaczynający się od T, potem zawierający dowolną literę, a na końcu samogłoskę łacińską

Negację zbioru tworzymy poprzez znak `^`:

`[^0-9]` linia zawierająca znak, który nie jest cyfrą
`[^-0-9]` linia zawierająca znak, który nie jest cyfrą ani minusem

Gdy znak specjalny pojawi się w złym miejscu, często może być on zinterpretowany dosłownie. I tak na przykład:

`[-0-9]` linia zawierająca znak, który jest cyfrą lub minusem
`[0-9-]` to samo, co wyżej
`[]0-9]` linia zawierająca znak, który jest cyfrą albo]
`[0-9]]` linia zawierająca cyfrę, a po niej]
`[0-9-z]` linia zawierająca cyfrę lub któryś ze znaków: `-`, `z`

Niektóre grupy znaków są tak popularne, że wprowadzono specjalny symbol:

`.` dowolny znak
`\s` dowolny biały znak
`\d` dowolna cyfra
`\w` dowolny znak słowny, równoważne `[a-zA-Z0-9_]`

Powyższe zbiory mogą być zanegowane poprzez wpisanie wielkiej litery: `\S`, `\D`, `\W`.

Istnieją również alternatywne oznaczenia dla zbiorów - klasy znaków `POSIX`:

`[:alnum:]` `A-Za-z0-9`
`[:alpha:]` `A-Za-z`
`[:blank:]` spacja lub tabulator
`[:cntrl:]` znaki kontrolne
`[:digit:]` `0-9`
`[:graph:]` znaki graficzne, czyli znaki o kodach `ACSII` 33-126
`[:lower:]` `a-z`
`[:print:]` znaki drukowalne (znaki graficzne + spacja)
`[:space:]` białe znaki
`[:upper:]` `A-Z`
`[:xdigit:]` `0-9A-Fa-f`

Uwaga! Te klasy wymagają ujęcia w dodatkową parę nawiasów kwadratowych, np.:

`[[:digit:]]` zbiór oznaczający dowolną cyfrę
`[[:alnum:]_]` zbiór oznaczający dowolną literę, cyfrę bądź podkreślnik

Znaki pozycjonujące

W przeciwieństwie do globów, `grep` nie musi dopasować się do całego słowa czy linii by stwierdzić dopasowanie. Wystarczy, że dowolny fragment linii będzie pasować. Jednakże, takie punkty specjalne, jak np. początek linii czy słowa można określić wprost wewnątrz wzorca.

Znaki pozycjonujące to:

`^` początek linii
`$` koniec linii
`b` granica słowa (początek lub koniec)
`[[:<:]]` początek słowa
`[[:>:]]` koniec słowa

Przykłady:

`\brok\b` Słowo „rok”
`rok` Fragment „rok”, być może jako fragment dłuższego słowa, np. „krok”.
`^A` „A” na początku linii
`A$` „A” na końcu linii

Znaki modyfikujące

Znaki modyfikujące pozwalają na różne operacje z użyciem poprzedzającego go znaku lub całego fragmentu wzorca.

`\` zmienia znaczenie następnego znaku ze specjalnego na zwykły i na odwrót
`*` poprzednik (znak lub wyrażenie w nawiasach) występuje 0 lub więcej razy
`+` poprzednik występuje 1 lub więcej razy
`?` poprzednik występuje raz lub wcale
`{n}` poprzednik występuje dokładnie n razy
`{n,}` poprzednik występuje co najmniej n razy
`{n,m}` poprzednik występuje co najmniej n razy, ale nie więcej niż m razy

Co więcej, można nakazać danej sekwencji znaków pojawiać się dokładnie określoną liczbę razy przy pomocy wstawienia przedziału wewnątrz `{` i `}`.

Przykłady:

`\\` dowolna linia z backslashem
`^\*` dowolna linia zaczynająca się od gwiazdki
`^A*` dowolna linia
`^AA*` dowolna linia zaczynająca się od A
`^A+` to samo, co wyżej
`^A?B` dowolna linia zaczynająca się od B lub od AB
`^A{4}B` dowolna linia zaczynająca się od AAAAB
`^A{4,8}B` dowolna linia zaczynająca się od 4, 5, 6, 7 lub 8 A, poprzedzających B
`\balpha\b.*\bbeta\b` linia zawierające słowa „alpha” i „beta” w tej właśnie kolejności

Grupowanie

W przeciwieństwie do globów, wyrażenia regularne można grupować. Powyżej wymienione znaki modyfikujące tyczą się wtedy całej grupy a nie pojedynczego znaku.

Grupy tworzy się za pomocą pary nawiasów `(`, `)` Na przykład:

`[A-Z][a-z]` Para liter, gdzie pierwsza jest duża a druga mała
`([A-Z][a-z])*` Dowolnej długości sekwencja par takich jak wyżej
`^(\b[A-Za-z]+\b ?)+$` Linia zawierająca jedynie słowa oddzielone pojedynczą spacją

Proszę zwrócić uwagę, że jeśli wzorzec jest powtarzany, za każdym razem może się dopasować do innej sekwencji znaków. Można jednak odwołać się do dopasowanego tekstu. Każda grupa zapamiętuje dopasowany fragment tekstu w slotach pamięci od `\1` od `\9`. Slot `\`n jest wykorzystywany przez n-tą grupę we wzorcu, licząc od lewej.

Przykładowo, pięcioliterowych palindromów (takich jak np. „radar") możemy szukać przez następujące wyrażenie regularne:

```
\b([a-z])([a-z])[a-z]\2\1\b
```

Alternatywa

Znak specjalny `|` oznacz alternatywę. W pierwszej kolejności dopasowywany będzie wzorzec po lewej, a jeśli się to nie powiedzie, to ten po prawej. Znaków `|` może być więcej, tworząc więcej niż 2 alternatywy.

W przypadku wyszukiwania słów „alfa" lub „beta" możemy postąpić tak, jak poniżej:

```
grep '\balfa\b|\bbeta\b' tekst.in
```

Często wszystkie elementy alternatywy umieszcza się w jeden wspólny nawias, by zaznaczyć końce tej alternatywy:

```
grep '(\balfa\b|\bbeta\b)' tekst.in
```