💻 Examen 103 - Comandos GNU y Unix

103.7 Buscar en archivos de texto usando expresiones regulares

Teoría

Las expresiones regulares (a menudo abreviadas como regex o regexp) son secuencias de caracteres que definen un patrón de búsqueda. Son un lenguaje conciso y flexible para describir patrones en texto. Se utilizan ampliamente en herramientas de línea de comandos, lenguajes de programación y editores de texto para tareas de búsqueda, reemplazo y validación.

Conceptos Clave y Sintaxis Básica de Regex:

Una expresión regular puede contener caracteres literales (que coinciden consigo mismos) y metacaracteres (que tienen un significado especial).

1. **Caracteres Literales:** La mayoría de los caracteres (letras, números, etc.) coinciden con ellos mismos. `a` coincide con "a", `123` coincide con "123".
2. **Metacaracteres Comunes:**
    - **`.` (Punto):** Coincide con cualquier carácter individual, *excepto* una nueva línea.
        - `a.c` coincide con "abc", "aec", "a1c", etc.
    - **`^` (Acento circunflejo):** Coincide con el inicio de la línea.
        - `^inicio` coincide con líneas que *empiezan* con "inicio".
    - **`$` (Signo de dólar):** Coincide con el final de la línea.
        - `fin$` coincide con líneas que *terminan* con "fin".
        - `^$` coincide con líneas vacías.
    - **`*` (Asterisco):** Coincide con **cero o más** ocurrencias del carácter o grupo inmediatamente anterior.
        - `a*b` coincide con "b", "ab", "aab", "aaab", etc.
        - `ab*c` coincide con "ac", "abc", "abbc", "abbbc", etc.
    - **`+` (Signo más):** Coincide con **una o más** ocurrencias del carácter o grupo inmediatamente anterior. (A menudo requiere usar `grep -E` o `egrep` para el *modo extendido* de regex).
        - `a+b` coincide con "ab", "aab", "aaab", etc., pero *no* con "b".
    - **`?` (Signo de interrogación):** Coincide con **cero o una** ocurrencia del carácter o grupo inmediatamente anterior. (A menudo requiere usar `grep -E` o `egrep`).
        - `colou?r` coincide con "color" y "colour".
    - **`|` (Barra vertical):** Actúa como un operador OR lógico. Coincide con el patrón a la izquierda O el patrón a la derecha. (A menudo requiere `grep -E`).
        - `cat|dog` coincide con "cat" o "dog".
    - **`( )` (Paréntesis):** Agrupan partes de una expresión regular para aplicar cuantificadores o como referencia posterior. (A menudo requiere `grep -E`).

        - `(ab)+` coincide con "ab", "abab", "ababab", etc.
    - **`[]` (Corchetes):** Definen un **conjunto de caracteres**. Coincide con *cualquier único* carácter dentro del conjunto.
        - `[aeiou]` coincide con cualquier vocal minúscula.
        - `[0-9]` coincide con cualquier dígito (equivalente a `\d` en regex extendidas).
        - `[a-zA-Z]` coincide con cualquier letra mayúscula o minúscula.
    - **`[^]` (Corchetes con acento circunflejo inicial):** Definen un **conjunto de caracteres negado**. Coincide con *cualquier único* carácter *que no* esté dentro del conjunto.
        - `[^0-9]` coincide con cualquier carácter que no sea un dígito.
        - `[^aeiou]` coincide con cualquier carácter que no sea una vocal minúscula.
    - **`{}` (Llaves):** Son **cuantificadores** específicos. (A menudo requiere `grep -E`).
        - `a{3}` coincide exactamente con "aaa".
        - `a{3,}` coincide con "aaa" o más 'a' (tres o más).
        - `a{3,5}` coincide con "aaa", "aaaa", o "aaaaa" (entre 3 y 5).
3. **Escape de Metacaracteres (`\`):** Si quieres que un metacaracter coincida con su propio valor literal (ej: buscar un punto `.`), debes "escaparlo" con una barra invertida (`\`).
    - `\.` coincide con un punto literal.
    - `\*` coincide con un asterisco literal.
    - `\\` coincide con una barra invertida literal.
4. **Clases de Caracteres Comunes (a menudo con `grep -E`):**
    - `\d`: Coincide con cualquier dígito [0-9].
    - `\s`: Coincide con cualquier carácter de espacio en blanco (espacio, tabulación, nueva línea, etc.).
    - `\w`: Coincide con cualquier carácter de palabra (letra, número o guion bajo _).
    - `\b`: Coincide con un límite de palabra. `\bcat\b` coincide con la palabra completa "cat", no con "catalogo" o "ubicacion".

La Herramienta Principal: `grep`

`grep` es la herramienta estándar de línea de comandos para buscar patrones (regex) en texto.

- `grep [opciones] 'patrón_regex' [archivo...]`
- `grep 'patron' archivo`: Busca el patrón en el archivo.
- `grep -v 'patron' archivo`: Invierte la coincidencia (muestra líneas que *no* coinciden).
- `grep -i 'patron' archivo`: Ignora mayúsculas/minúsculas.
- `grep -n 'patron' archivo`: Muestra el número de línea donde se encontró la coincidencia.
- `grep -c 'patron' archivo`: Cuenta el número de líneas que coinciden.
- `grep -l 'patron' archivo*`: Lista solo los nombres de los archivos que contienen una coincidencia.

- `grep -r 'patron' directorio`: Busca recursivamente en directorios.
- `grep -E 'patrón_regex_extendido' archivo`: Habilita el soporte para expresiones regulares extendidas (permite usar `+`, `?`, `|`, `()` sin escaparlos). Es lo mismo que el comando `egrep`.
- `grep -F 'cadena_literal' archivo`: Trata el patrón como una cadena fija, no como regex. Es más rápido si solo buscas texto literal. Es lo mismo que el comando `fgrep`.

Otras Herramientas con Regex:

- **`sed`:** Usa regex en su comando `s/patron/reemplazo/` para sustituir texto.
- **`awk`:** Usa regex en condiciones y para procesar campos.
- **`find`:** La opción `-regex` permite buscar archivos cuyo nombre coincide con una regex.
- **Editores de texto:** `vim`, `nano` (con búsqueda), `emacs` y otros editores soportan búsqueda y reemplazo con regex.

Dominar regex requiere práctica, pero es una inversión de tiempo que vale la pena para cualquier persona que trabaje con datos textuales en Linux.