Конспект лекции «Регулярные выражения»

Причина появления регулярных выражений - потребность в средстве поиска нужной строки в тексте одной командой.

Впервые появились в текстовом редакторе QED в 1960х (Кен Томпсон).

Таким образом, регулярные выражения – это такой формальный язык, который осуществляет манипуляции со строками в тексте (~ шаблоны для работы с подтекстом), другими словами, система поиска текстовых фрагментов, основанная на записи специальным образом шаблонов поиска.

Пример:

есть директории standart, post, nasty, alibaba, необходимо вывести только те, которые содержат подстроку «st»;

решение ls *st*.

Наборы символов

обычные (литералы)	метасимволы
- любые символы кроме специальных (но их можно заэкранировать «\» либо вставить между \Q и \E)	[] \ ^ $. | ? * + () {}

Поддерживаемые операции:

1. сцепление, напр., {“ab”, “cd”}{“12”,”54”} -> “ab12”,”54cd”,…;
2. дизъюнкция {“ab”, “cd”}|{“12”,”54”};
3. замыкание {“ab”, “cd”}*.

Понятие **символьных классов**

– это набор символов, заключенных в [], что означает, что на данном месте в строке может стоять один из перечисленных символов,

например, [1234], [Ss],

[a-z], [A-Za-z] – диапазоны символов,

[-+=0-9] – любая цифра или знаки -+= (специальные символы теряют свой смысл в начале и конце диапазона),

[^abc] – все символы кроме заданных,

\d – любая цифра,

\D – все, кроме цифры,

\s – пробел,

\S – все, кроме пробельного символа,

\w = [A-Za-z0-9_],

\W = [^A-Za-z0-9_].

Способы задания **позиции внутри строк**:

^ - начало строки,

$ - конец строки,

\b – граница слова,

\B – не граница слова,

\G – предыдущий успешный поиск,

напр., \Ga -> aaaa aaaa -> остановится на 4ой позиции, т.е. там, где не нашел более а.

Квантификация

- для поиска конкретного числа повторений последовательностей символов (т.е. может относится к одному символу, символьному классу или группе).
{n} – ровно n раз,
{m,n} – от m до n раз включительно,
{m,} – не менее m раз,
{,n} – не более n раз,
* = {0,},
+ = {1,},
? = {0,1}.

Жадная квантификация * + {n,} – напр., <.*> будет искать максимальное вхождение соответствующее выражению.

Ленивая квантификация *? +? {n,}? – <.*?> будет искать наиболее короткие строки.

Ревнивая квантификация (сверхжадная).

Группировка

- используется для определения области действия с помощью (), после которых можно воспользоваться квантификацией, напр., (abc)*,

также можно применять как повторное использование ранее найденных групп символов (обозначаются от \1 до \9),

(?abc) – такая группа сохранятся не будет, номер присваиваться ей не будет.

Пример регулярного выражения MAC-адреса:
/^(?[0-9A-Fa-f]{2}:){5}[0-9A-Fa-f]{2}$/

Д/з №1: записать регулярное выражение для MAC-адреса короче.

Д/з №2: написать команду, которая будет искать все URL-адреса в файле. URL выглядит как www.domain.domain2. При этом www может отсутствовать и domain2 - двух или трехбуквенный.

Д/з №3: написать regexp для разбора ip-адреса. Написать надо именно команду (cat file | grep...)

Существует по крайней мере **три синтаксиса регулярных выражений:**

POSIX basic POSIX extended Perl

1. **Базовые регулярные выражения**

. – любой символ, если не внутри [];
[] – символьный класс;
^ – начало строки;
$ – конец строки;
* – предыдущий элемент любое количество раз;
\(\) – подвыражение;

\{n, m\} – предыдущий элемент от n до m раз.

Символы () и {} без \ ищет сами скобки!

2. Расширенные регулярные выражения

Добавлено:
? = {0,1};
+ = {1,};

Скобки () {} в качестве метасимволов теперь не экранируются, для поиска самих скобок требуется приписать "\": \(\)

3. Выражения в стиле Perl

Добавляет к ERE:
ленивые квантификаторы – *?, +?, ??
сверхжадные квантификаторы – *+, ++, ?+
сокращенные записи символьных классов - \w, \W, \s, \S

Команда GREP

grep [options] pattern [file…] - поиск строки, соответсвующей pattern (без ключей просто печатает найденные строки), возвращает 1, если что-то нашел, и 0, если ничего не нашел.
Опция «–o» - подсчет количества соотвествующих строк.
Опция «–l» - вывод имен файлов, содержащих строки, соотвествующие шаблону.
напр., grep –l '[…]' /etc/*
Опция «–r» - то же самое, но включая подкаталоги:
grep –l –r '[…]' /etc/hosts
Опция «–v» - отрицание условия поиска.

Команда SED

sed [options]… {script} [inputfile]… - читает строку, выводит, исправляя ее в соответствии со скриптом, затем читает следующую строку и так далее (о каждой предыдущей строке сразу забывает), имеет два буфера для хранения данных: pattern space – основной и hold space – дополнительный (для хранения промежуточных состояний).

Способы передачи скрипта:
sed –e script [inputfile];
sed -f [scriptfile] [inputfile];
sed [options] script [inputfile];

опция –i позволяет записать результат поверх исходного файла,
'=' – добавляет в начало строки ее номер,
'd' – очищение строки (весь файл будет пуст),
'3d' – указание номера строки, к которой применяется операция,
'2,4d' – указание диапазона строк,
/{/,/}/d – адресация по регулярному выражению (исчезнут все блоки вида {…}) (регулярное выражение должно быть окружено косыми чертами //).
s/abc/dfg/ - замена выражения abc на dfg (будет заменено только первое найденное соотвествующее выражение в текущей строке),

s/abc/dfg/g – замена во всей строке,
& - подставить найденную подстроку,
a / text – вставляет text ниже текущей сроки,
i / text – вставляет text выше текущей сроки,
c / text – вставляет text вместо текущей сроки,

Использование Hold Space

Пример: sed –e ‘1!G;h;$!d’ forward.txt > backward.txt

1!G – для каждой строки, кроме первой, дописывает содержимое hold space в конец pattern space,

h – копирует содержимое pattern space в hold space,

$!d – применяет команду d ко всем строкам, кроме последней.

Д/з№4: Имя, Фамилия, Телефон -- телефонная книжка в csv. Преобразовать в html, который запустится в браузере. Только с помощью sed!

Д/з№5: Файл file.c, вывести все хедеры (только имена самих библиотек).

Д/з№6: Утилита /sbin/ifconfig, выводит названия интерфейсов и их параметры. Все IP-адреса всех интерфейсов заменить на xxx.x.x.x, каждый икс соответствует одной цифре в IP-адресе. Разделить интерфейсы строкой дефисов.