sed, awk, regex

cut and pasted from the net

rev. 2014-02-24-03

sed description

- pattern a text → add to output
- address s /regex/replacement/
- address d → delete line
- delete lines 1-10: `sed -e '1,10d'`
- delete comments: `sed -e '/^#/d'`
- print only matching:

```
sed -n -e '/regexp/p'
```

- convert Unix to DOS:

```
sed -e 's/$/\r/' myunix.txt > mydos.txt
```

awk

- Special-purpose language for line-oriented pattern processing
- pattern {action}
- action =
  - if (conditional) *statement* else *statement*
  - while (conditional) statement
  - break
  - continue
  - variable=expression
  - print expression-list

Examples (1)

```
$ awk '{ print $0 }' /etc/passwd
$ awk '{ print "" }' /etc/passwd
$ awk '{ print "hiya" }' /etc/passwd
$ awl -f file.awk /etc/passwd

file.awk:
BEGIN  { FS=":" }
END    { print “xxxxxxxxxxxxxxxxxxx” }
/user/ { print }
/[0-9]+.[0-9]*/ { print }
{ print $1 }
```

Examples (2)

- Print first two fields in opposite order

  awk '{ print $2, $1 }' file
- Print lines longer than 72 characters:

  awk 'length > 72' file
- Print length of string in 2nd column

  awk '{print length($2)}' file
- Add up first column, print sum and average

```
{ s += $1 }
END {
    print "sum is",s," average is", s/NR
}
```

Examples (3)

- Print fields in reverse order

  awk '{ for (i=NF; i > 0; --i) print $i }' file

- Print the last line

  {line = $0}

  END {print line}

- Print the total number of lines of word "Pat"

  /Pat/ {nlines = nlines + 1}

  END {print nlines}

Examples (4)

- Print all lines between start/stop pairs

  awk '/start/, /stop/' file
- Print all lines whose first field is different from previous one

  awk '$1 != prev { print; prev = $1 }' file
- Print column 3 if column 1 > column 2

  awk '$1 > $2 {print $3}' file
- Print line if column 3 > column 2

  awk '$3 > $2' file

Examples (5)

- awk '$3 > $1 {print i + "1"; i++}' file
- awk '{print NR, $1}' file
- awk '{$2 = ""; print}' file
- yes | head -28 | awk '{ print "hi" }'
- yes | head -90 | \
  awk '{printf("hi00%2.0f n", NR+9)}'
- yes | head -4 | awk '{print rand()}'
- yes|head -40|awk '{print int(100*rand())%5}'
- { for (i = 1; i <= NF; i=i+1)
  if ($i < 0) $i = -$i print}

What Is a Regular Expression?

- A regular expression (regex) describes a set of possible input strings.
- Regular expressions descend from a fundamental concept in Computer Science called finite automata theory
- Regular expressions are endemic to Unix
  - vi, ed, sed, and emacs
  - awk, tcl, perl and Python
  - grep, egrep, fgrep
  - compilers
- The simplest regular expressions are a string of literal characters to match.
- The string matches the regular expression if it contains the substring

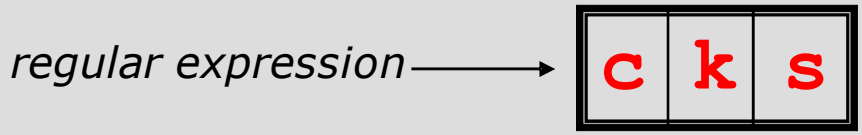
Regular Expressions

Fundamentals

Match the specified character unless it is a ...

.	Match any character (except EOL)
[character class]	Match the characters in character class.
[start-end]	start to end
[^character class]	Match anything except the character class.
$	Match the end of the line
^	Match the beginning of the line
*	Match the preceding expression zero or more times
?	Match the preceding zero or one time
|	Match the left hand side OR the right side
(regexp)	Group the regular expression
\	Treat next character literally (not specially)

Regular Expressions




```
UNIX Tools rocks.
```

match

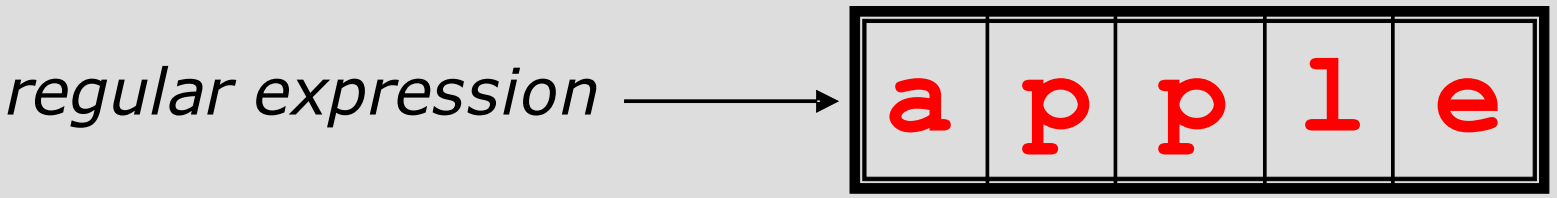
```
UNIX Tools sucks.
```

match

```
UNIX Tools is okay.
```

no match

Regular Expressions (con't)

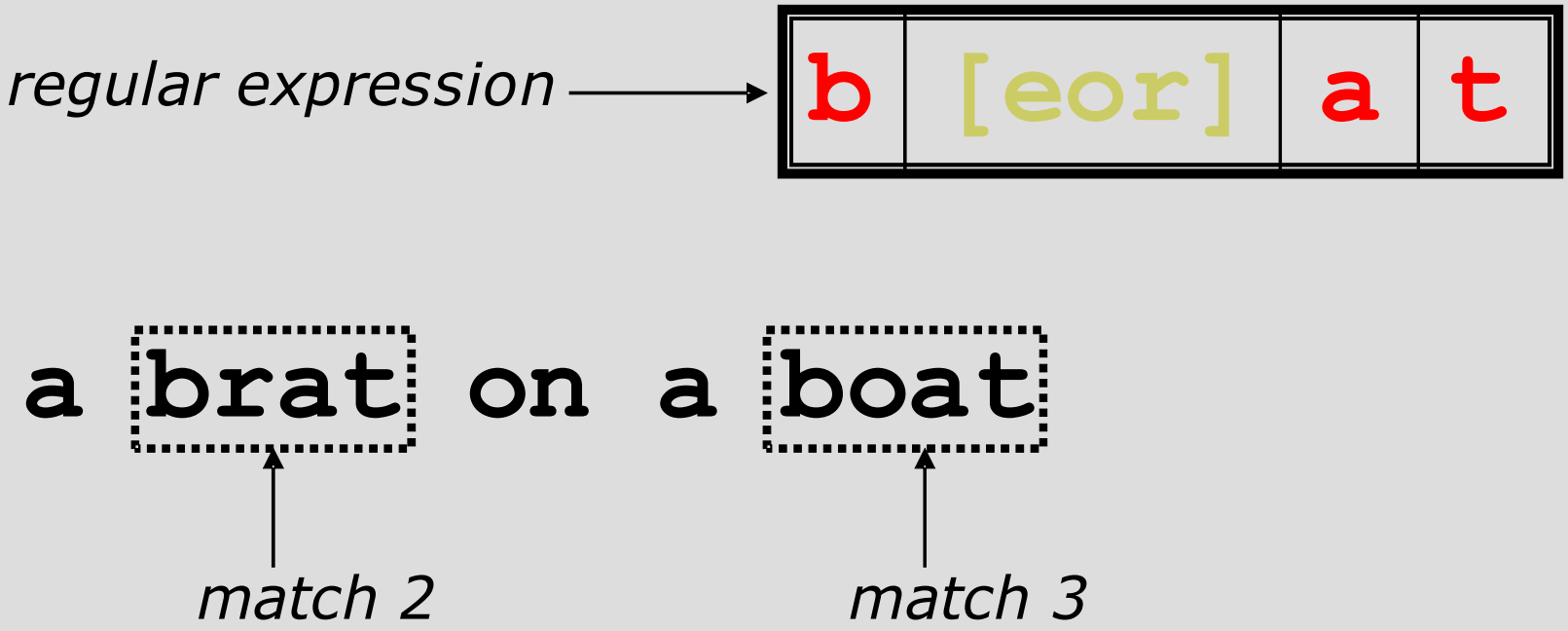


Scrapple from the apple.

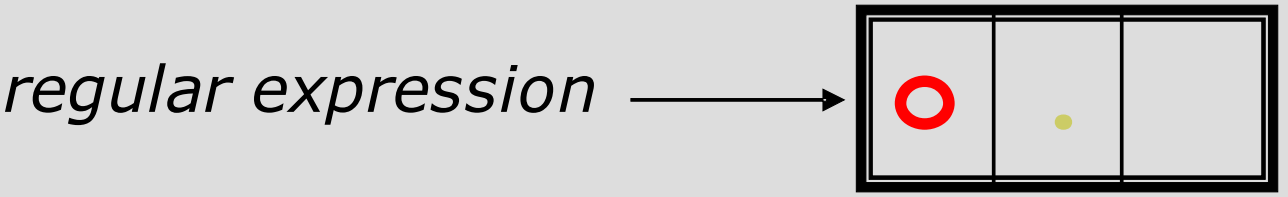
match 1

match 2

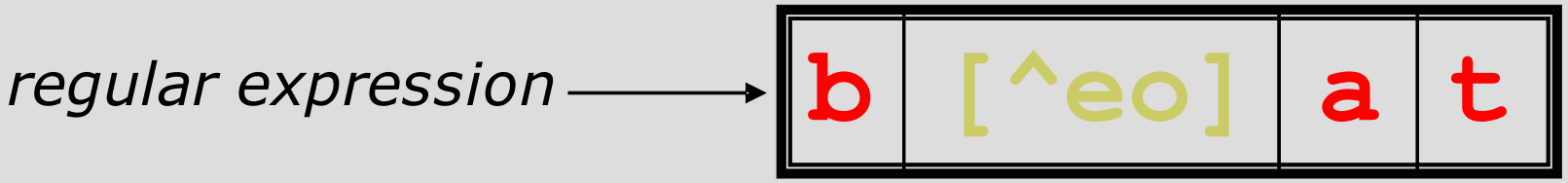
Character Classes



Character Classes (con't)



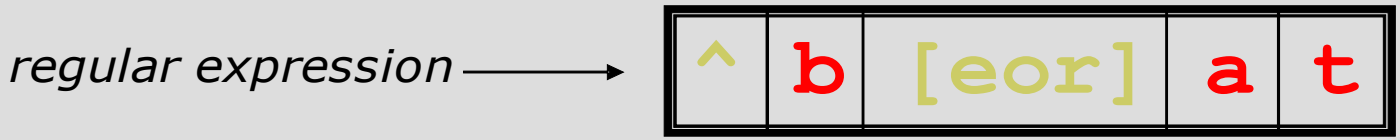
Negated Character Classes




```
beat a brat on a boat
```

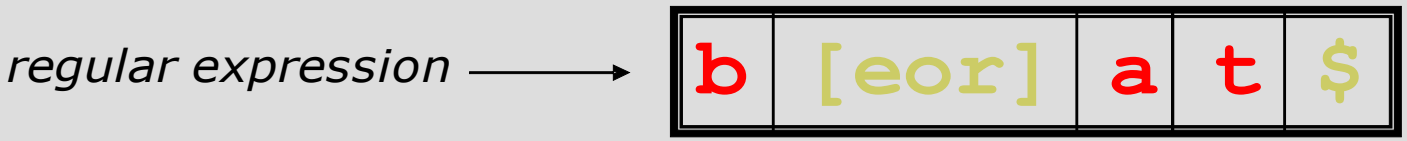
match

Anchors



beat a brat on a boat

match



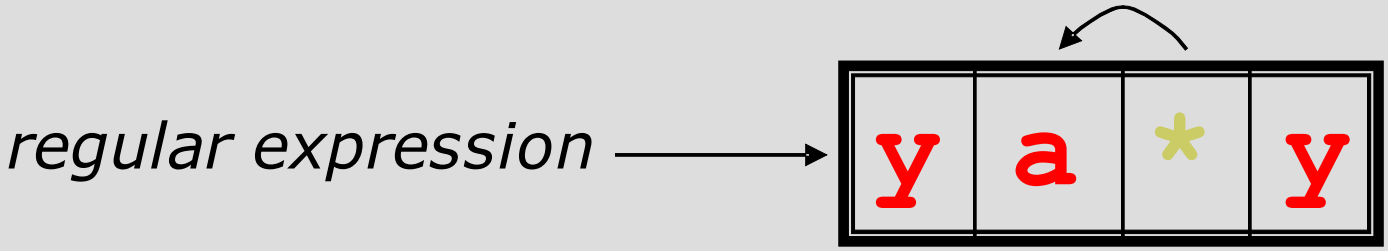
beat a brat on a boat

match

---

^word$ ^$

Repetitions

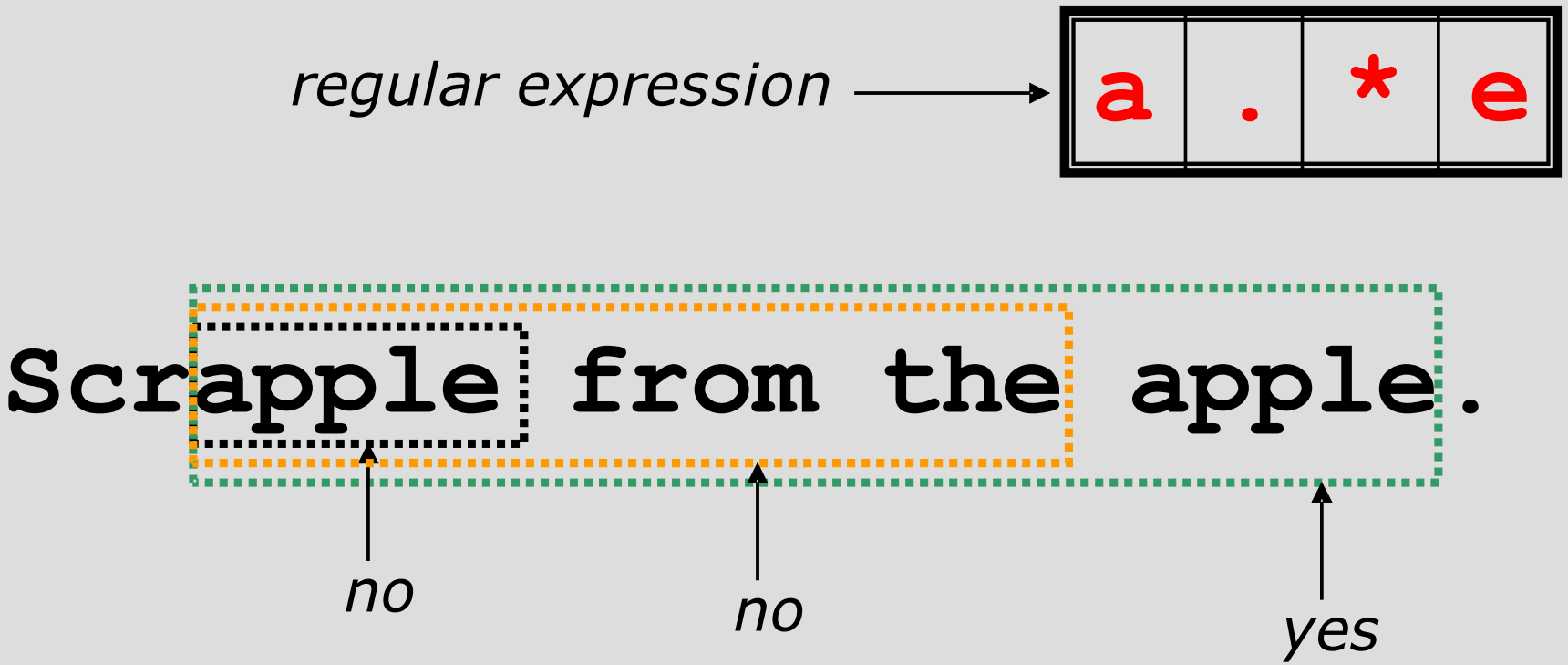



```
I got mail, yaaaaaaaaaaay!
```

match

.*

Match Length



Examples (1)

- Examples:

Match a line beginning with a space-padded line number and colon.

^[\t]*[0-9][0-9]*:

Match a name (various spellings)

(Tim Shelling)|(TJS)|(T\. Shelling)|(Timothy J\. Shelling)

Match if the line ends in a vowel or a number:

[0-9aeiou]$

Match if the line begins with anything but a vowel or a number:

^[^0-9aeiou]

Example (2)

- IP v4 Address (255.255.255.255)
- `\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\b`
  - `999.999.999.999`
- `\b(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.`
  `(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.`
  `(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.`
  `(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\b`
- ALAMAT@EMA.IL
  `\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b`
- YYYY-MM-DD (1900-01-01 sd. 2099-12-31)
  `(19|20)\d\d[- /.](0[1-9]|1[012])[- /.](0[1-9]|[12][0-9]|3[01])`