DEALING WITH DATA

SPRING 2019: INFO-GB.2346.30

PROFESSOR GUTHRIE COLLIN

TEACHING FELLOW AJINKYA WALIMBE

NYU | STERN

CLASS 8: REGULAR EXPRESSIONS ("REGEX")

APRIL 11, 2019

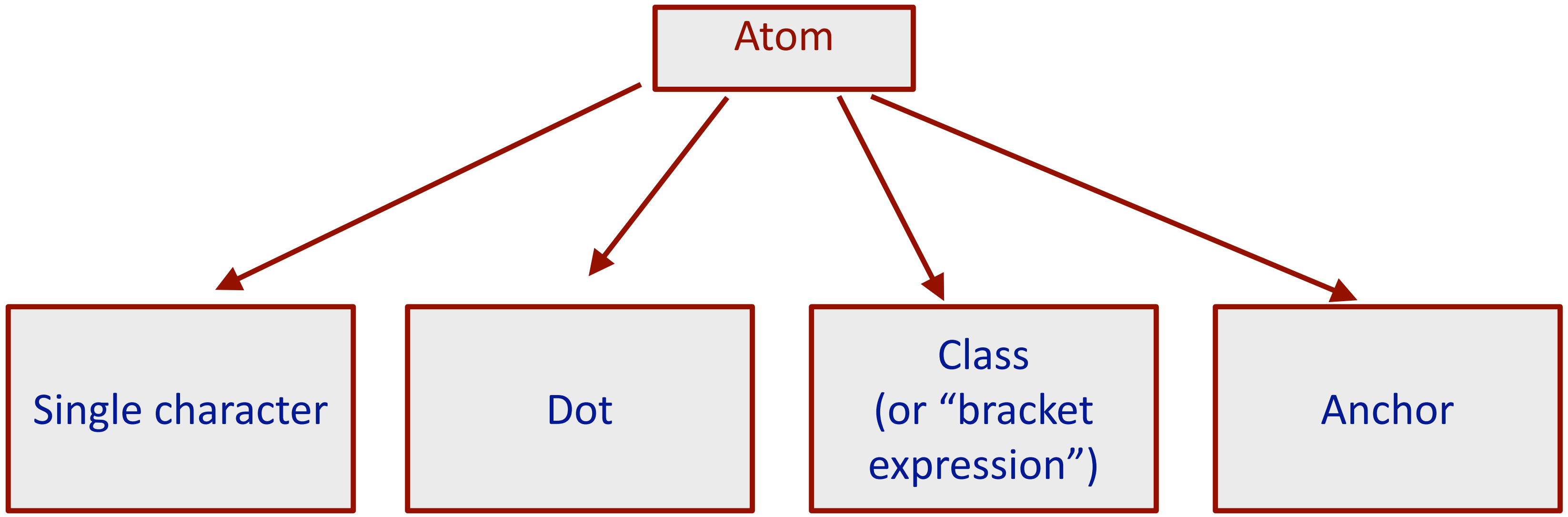
Regular Expressions

- A sequence of characters that forms a search pattern. Some of their uses:
  - pattern matching
  - string matching
  - “find-and-replace”
  - do something with matched pattern
  - validate data
  - parse data
  - etc.
- Typically made up from special characters called “metacharacters”

Grammar of Regex

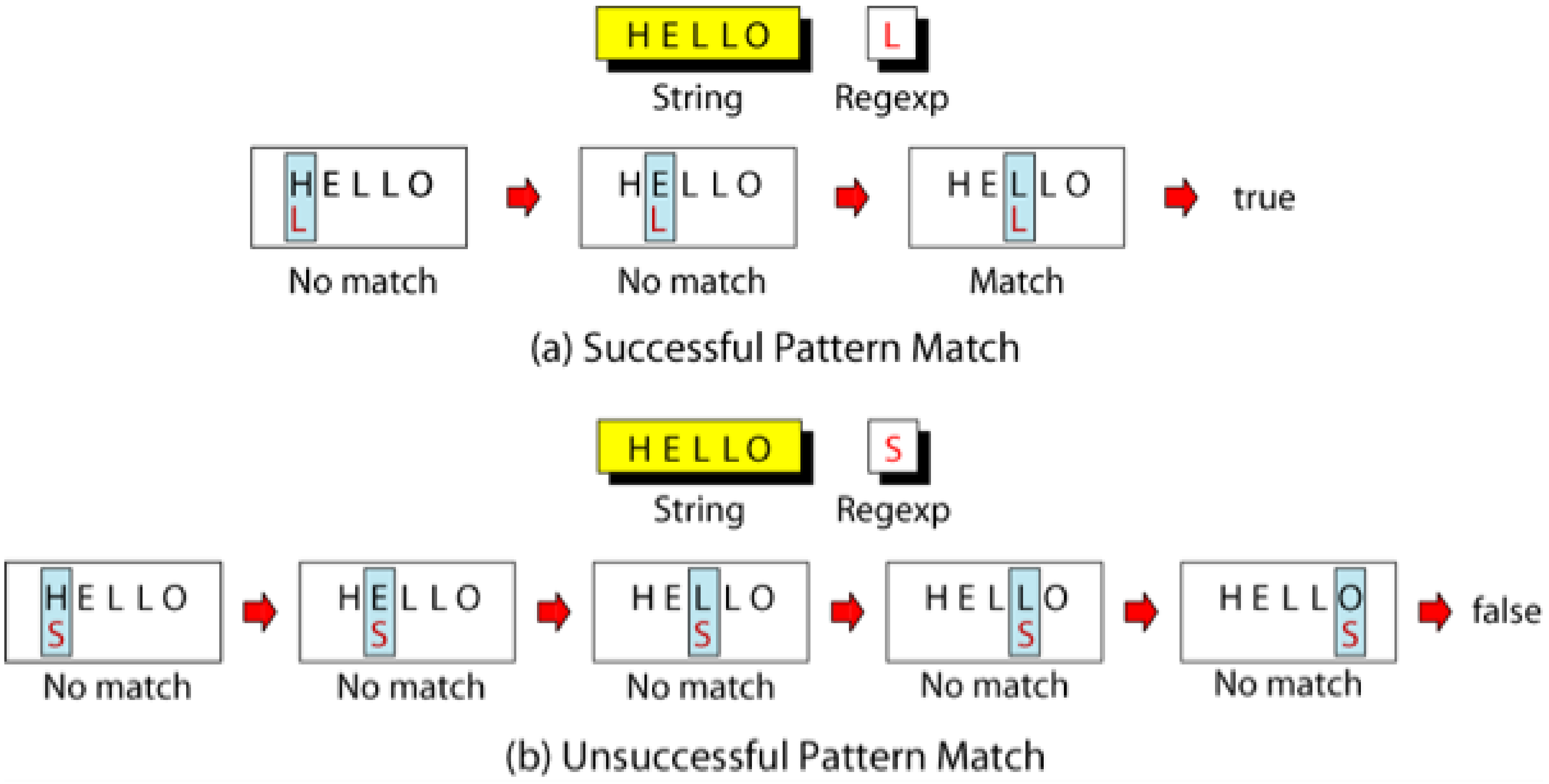
- regex = one or more non-empty branches separated by '|'
- branch = one or more pieces
- piece = atom followed by quantifier
- quantifier = *, +, ?or bound
- bound = atom{n}, atom{n,}, atom{m,n}
- atom = (regex) or () or '^,$' or \ followed by '^.[$()|*+?{\' or any character or bracket expression
- bracket-expression = a list of characters enclosed in brackets '[]'

Atoms


Atom
Single character
Dot
Class
(or “bracket
expression”)
Anchor

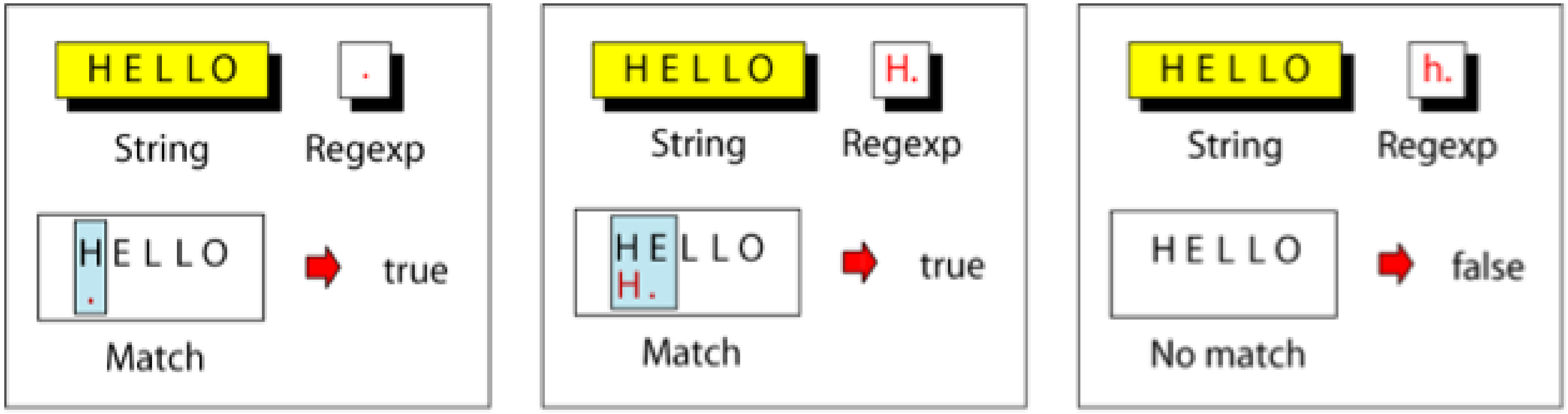
Single-Character

A single character atom matches itself



Dot

A dot atom matches any single character except for a new line character ("\n")

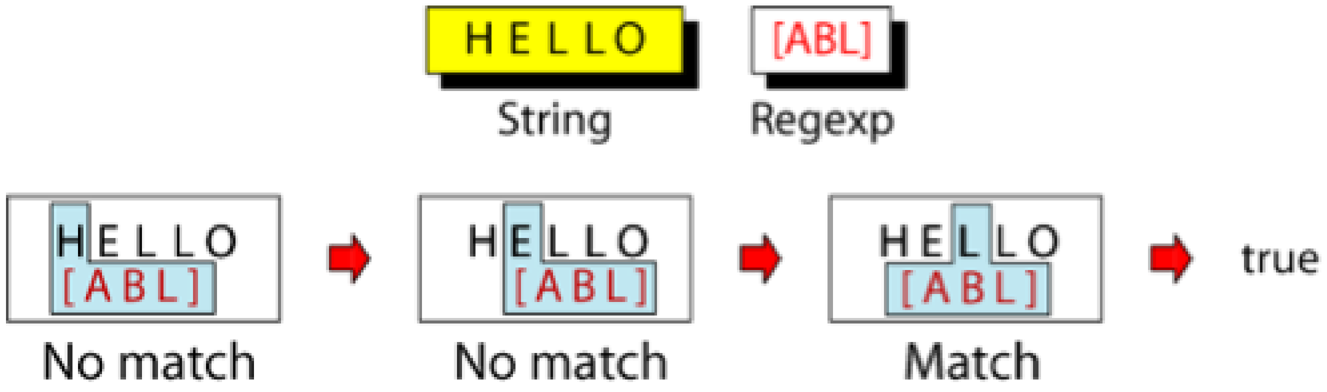


(a) Single-Character (b) Combination–True (c) Combination–False

Class / Bracket Expression

A class matches only one single character that can be any of the characters defined in a set (defined by square brackets []).

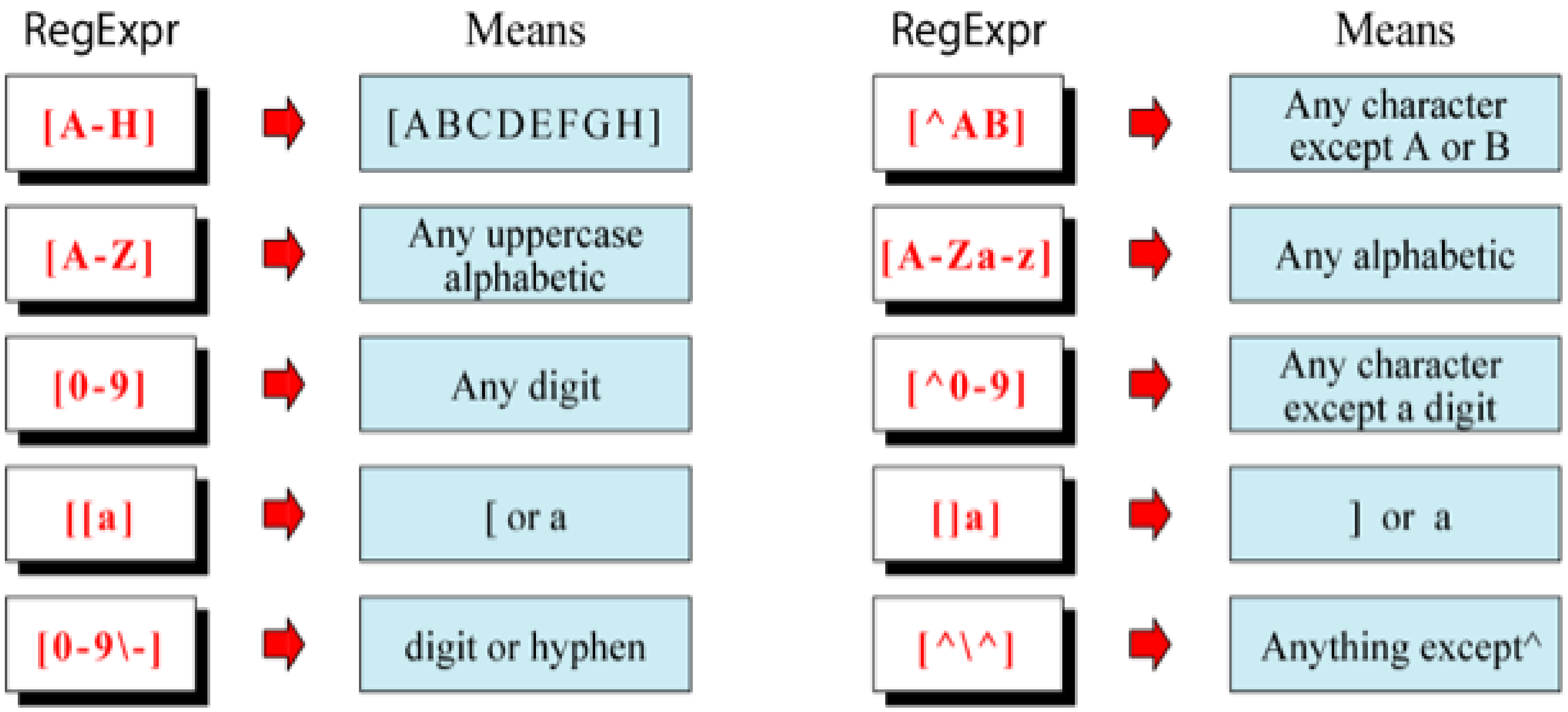
- Example: [ABL] matches either A, B, or L.



More about character classes

- Ranges can also be specified in character classes (-)
  - [1-9] is the same as [123456789]
  - [a-e] is equivalent to [abcde]
- You can also combine multiple ranges
  - [a-e1-9] is equivalent to [abcde123456789]
- You can also specify characters to be excluded from the set using the character (^)
  - [^0-9] matches any character other than a number.

Examples

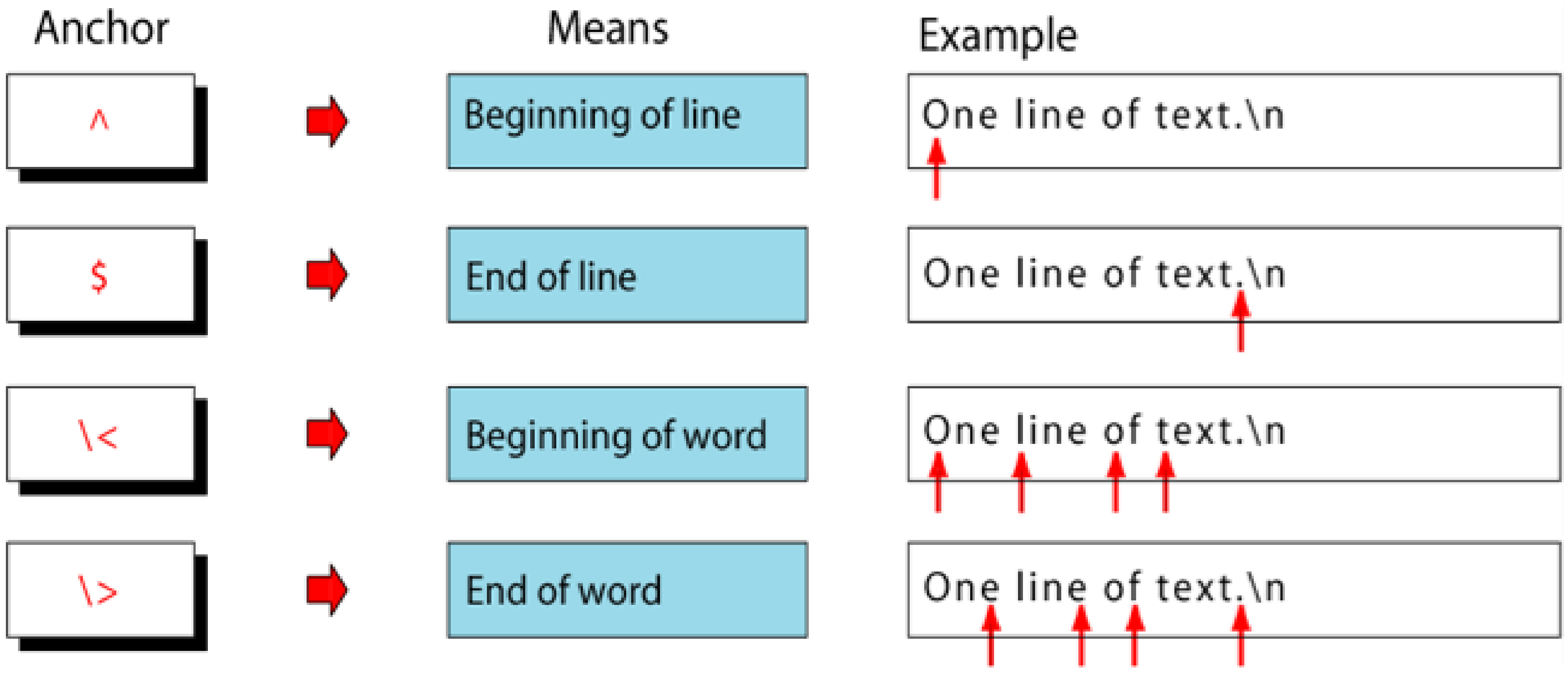


RegExpr	Means	RegExpr	Means
[A-H]	[ABCDEFGH]	[^AB]	Any character except A or B
[A-Z]	Any uppercase alphabetic	[A-Za-z]	Any alphabetic
[0-9]	Any digit	[^0-9]	Any character except a digit
[[a]	[or a	[]a]	] or a
[0-9\-]	digit or hyphen	[^\^]	Anything except^

Note: If you want to use special characters (e.g., -,^) as a literal, you need to escape them with a backslash

Anchors

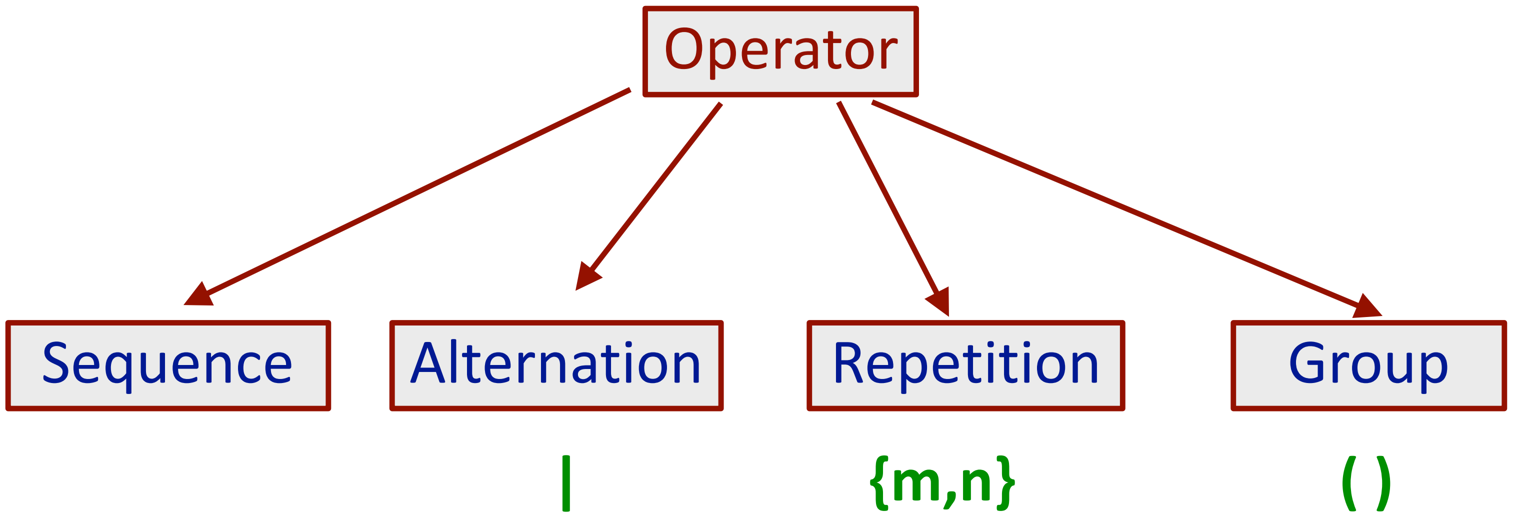
- An anchor atom specifies the position in the string where a match must occurs



Anchor	Means	Example
^	Beginning of line	One line of text.\n
$	End of line	One line of text.\n
\<	Beginning of word	One line of text.\n
\>	End of word	One line of text.\n

Operators

- An operator combines atoms



Sequence Operator

- A sequence of atoms

`dog`	matches the pattern "dog"
`a..b`	matches "a" , any two characters, and "b"
`[2-4][0-9]`	matches a number between 20 and 49
`[0-9][0-9]`	matches any two digits
`^$`	matches a blank line
`^.$`	matches a one-character line
`[0-9]-[0-9]`	matches two digits separated by a "-"

Alternation Operator (or): |

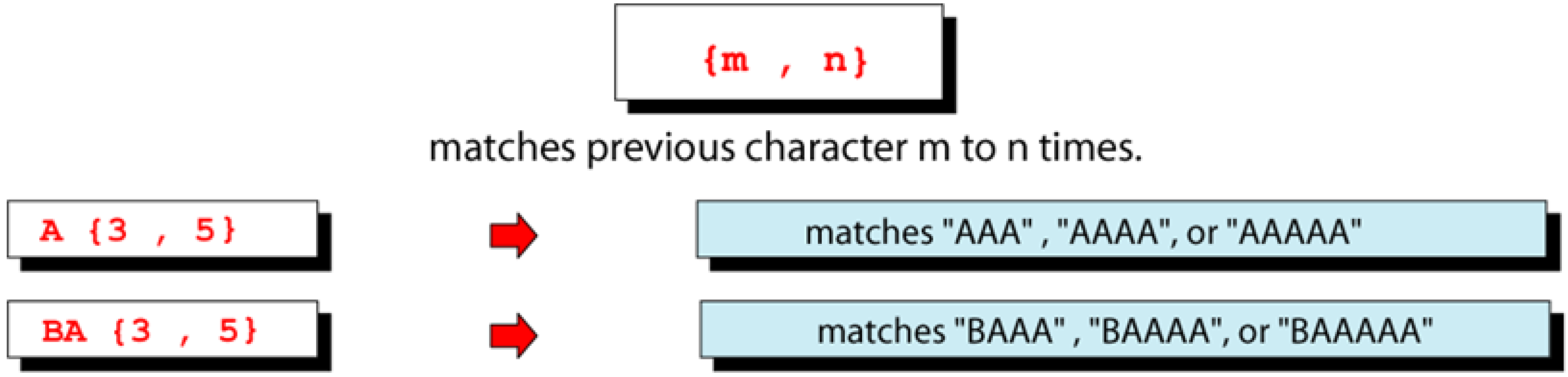
- An alternation operator is used to define one or more alternatives.

`UNIX|unix` → matches "UNIX" or "unix"

`Ms|Miss|Mrs` → matches "Ms" or "Miss" or "Mrs"

Repetition Operator: {}

- A repetition operator specifies that the atom or expression immediately before the repetition may be repeated.


{m , n}
matches previous character m to n times.
A {3 , 5}
matches "AAA" , "AAAA", or "AAAAA"
BA {3 , 5}
matches "BAAA" , "BAAAA", or "BAAAAA"

Basic Repetition Forms

Formats

`{m}`	matches previous atom exactly m times
`{m,}`	matches previous atom m times or more
`{, n}`	matches previous atom n times or less

Examples

`CA {5}`	CAAAAA
`CA {3,}`	CAAA, CAAAA, CAAAAA, …
`CA {,2}`	C, CA, CAA

Short Form Repetition Operators

Formats

`*`	special case: matches previous atom zero or more times
`+`	special case: matches previous atom one or more times
`?`	special case: matches previous atom 0 or one time only

Examples

`BA*`	B, BA, BAA, BAAA, BAAAA, . . .
`B.*`	B, BA . . . BZ, BAA . . . BZZ, BAAA . . . BZZZ, . . .
`.*`	zero or more characters
`.+`	one or more characters
`[0-9]?`	zero or one digit

Question: Repetition Operators

- Use short form repetition operators to represent the following regular expressions in basic repetition forms?

a{2,}

Question: Repetition Operators

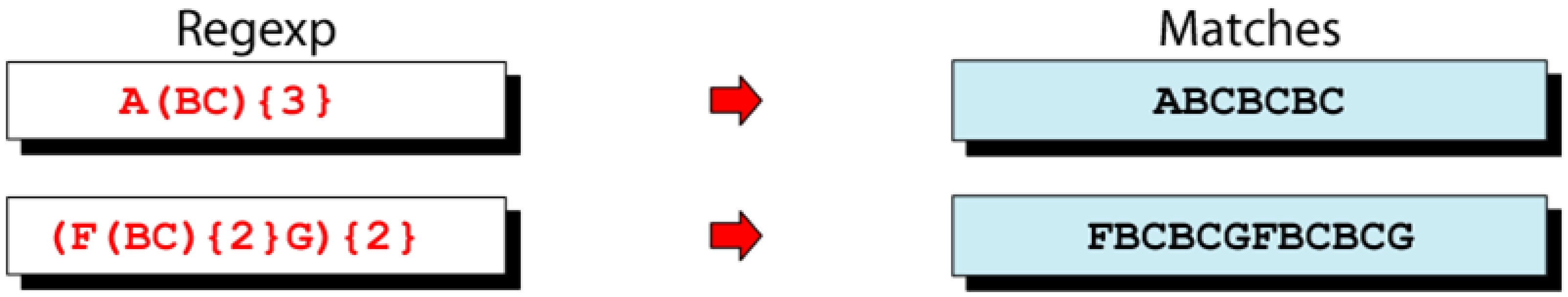
- Use short form repetition operators to represent the following regular expressions in basic repetition forms?

a{2,}

- Answer: aaa* or aa+

Group Operator

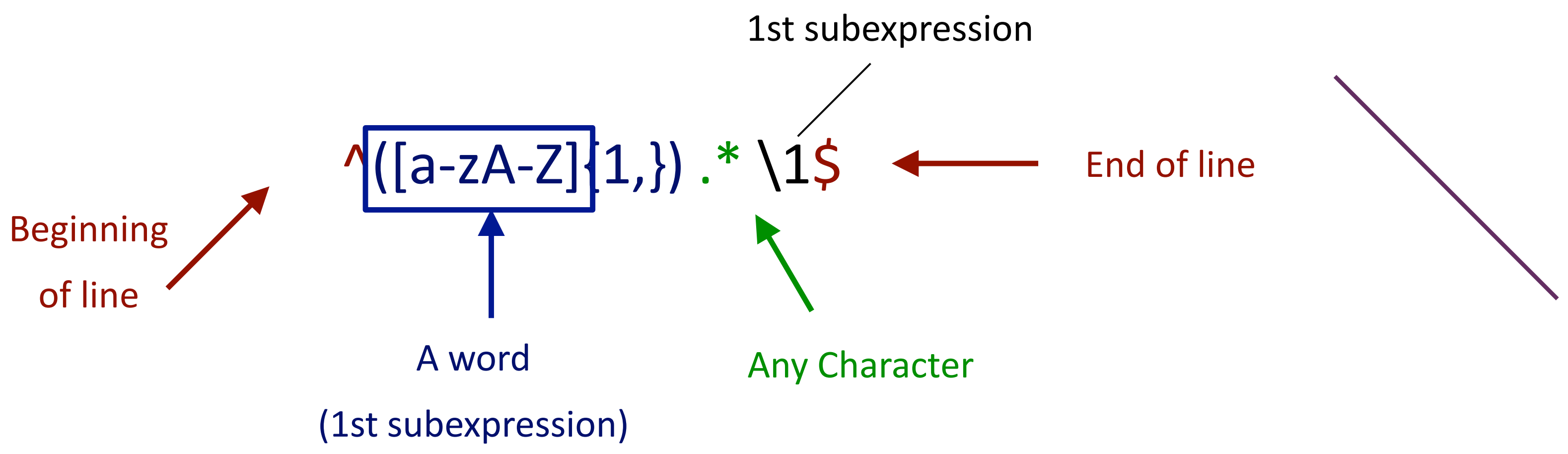
- In the group operator, when a group of characters is enclosed in parentheses, the next operator applies to the whole group, not only the previous characters.



Regexp		Matches
A(BC){3}	➡	ABCBCBC
(F(BC){2}G){2}	➡	FBCBCGFBCBCG

Grep: Backreference

- Sometimes it is handy to be able to refer to a match that was made earlier in a regex
- This is done with backreferences
  - \k is the backreference specifier, where k is a number
- Looks for nth subexpression
- For example, ***find if the first word of a line is the same as the last:***



Backreference Examples

- Find all lines in fields.txt that have a number in the form [0-9]*x[0-9]x[0-9]*, where x is a digit:
  - grep ‘([0-9])([0-9])\1’ fields.txt
- Find all numbers that have two consecutive same digits:
  - grep ‘([0-9])\1)’ fields.txt

Matching a float Number (Example)

- We want to match a signed float (decimal) number, i.e.,:
  - +12.34 or -1.457 or 1023.4568 etc.

Matching a float Number (Example)

- We want to match a signed float (decimal) number, i.e.,:
  - +12.34 or -1.457 or 1023.4568 etc.

- 1st step: we need at most a sign in the beginning : [+-]?

Matching a float Number (Example)

- We want to match a signed float (decimal) number, i.e.,:
  - +12.34 or -1.457 or 1023.4568 etc.

- 1st step: we need at most a sign in the beginning : [+-]?
- 2nd step: we need at least one digit: [0-9]+

Matching a float Number (Example)

- We want to match a signed float (decimal) number, i.e.,:
  - +12.34 or -1.457 or 1023.4568 etc.

- 1st step: we need at most a sign in the beginning : [+-]?
- 2nd step: we need at least one digit: [0-9]+
- 3rd step: we need one dot: \.

Matching a float Number (Example)

- We want to match a signed float (decimal) number, i.e.,:
  - +12.34 or -1.457 or 1023.4568 etc.

- 1st step: we need at most a sign in the beginning : [+-]?
- 2nd step: we need at least one digit: [0-9]+
- 3rd step: we need one dot: \.
- Final step: we need at least one digit: [0-9]+

Matching a float Number (Example)

- We want to match a signed float (decimal) number, i.e.,:
  - +12.34 or -1.457 or 1023.4568 etc.

- 1st step: we need at most a sign in the beginning : [+-]?
- 2nd step: we need at least one digit: [0-9]+
- 3rd step: we need one dot: \.
- Final step: we need at least one digit: [0-9]+
- Pattern that matches float numbers: [+-]?[0-9]+\.[0-9]+

Summary of what we learned and more

Meta	*Description*
.	Matches any single character. With bracket expressions, the dot character matches a literal dot. For example, a.c matches "abc", but [a.c] matches only "a", ".", or "c".
[]	A bracket expression. Matches a single character that is contained within the brackets. For example, [abc] matches "a", "b", or "c". [a-z] specifies a range which matches any lowercase letter from "a" to "z".
[^]	Matches a single character that is not contained within the brackets. For example, [^abc] matches any character other than "a", "b", or "c".
^	Matches the starting position within the string.
$	Matches the ending position of the string or the position just before a string-ending newline.
( )	Defines a marked subexpression.
\n	Matches what the nth marked subexpression matched, where n is a digit from 1 to 9.
*	Matches the preceding element zero or more times.
{m,n}	Matches the preceding element at least m and not more than n times.
?	Matches the preceding element zero or one time.
+	Matches the preceding element one or more times.
|	The alternation operator matches either the expression before or the expression after the operator

Summary of what we learned and more

Python	***ASCII***	***Description***
	[A-Za-z0-9]	Alphanumeric Characters
\w	[A-Za-z0-9_]	Alphanumeric characters plus "_"
\W	[^A-Za-z0-9_]	Non-word characters
	[A-Za-z]	Alphabetic characters
	[\t]	Matches space and tab
\d	[0-9]	Digits
\D	[^0-9]	Non - Digits
	[a-z]	Lower-case letters
	[A-Z]	Upper-case letters