جامعة المنوفية

MENOUFIA UNIVERSITY

FACULTY OF COMPUTERS AND INFORMATION

Fourth Year (Second Semester)

CS Dept., (CS 436)

Natural Language Processing NLP

Lecture Two

Dr. Hamdy M. Mousa

Regular Expression

- Regular Expression (RE)
  - (first developed by Kleene (1956)
  - is a formula in a special language that is used for specifying simple classes of strings.
  - is **a pattern** that matches some sequence in a text.
  - It is a mixture of:
    - characters or strings of text
    - special characters
    - groups or ranges
- Since common text-processing programs agree on most of the syntax of regular expressions,
  - all UNIX, Microsoft Word, and WordPerfect regular expressions.

Regular Expression

- Regular expression search requires a pattern that we want to search for.
  - a corpus of texts to search through.
- A regular expression search function will search through the corpus returning all texts that contain the pattern.
  - In an information retrieval (IR) system such as a web search engine, the texts might be entire documents or web pages.
  - In a word-processor, the texts might be individual words, or lines of a document.
  - A search can be designed to return all matches to a regular expression or only the first match.

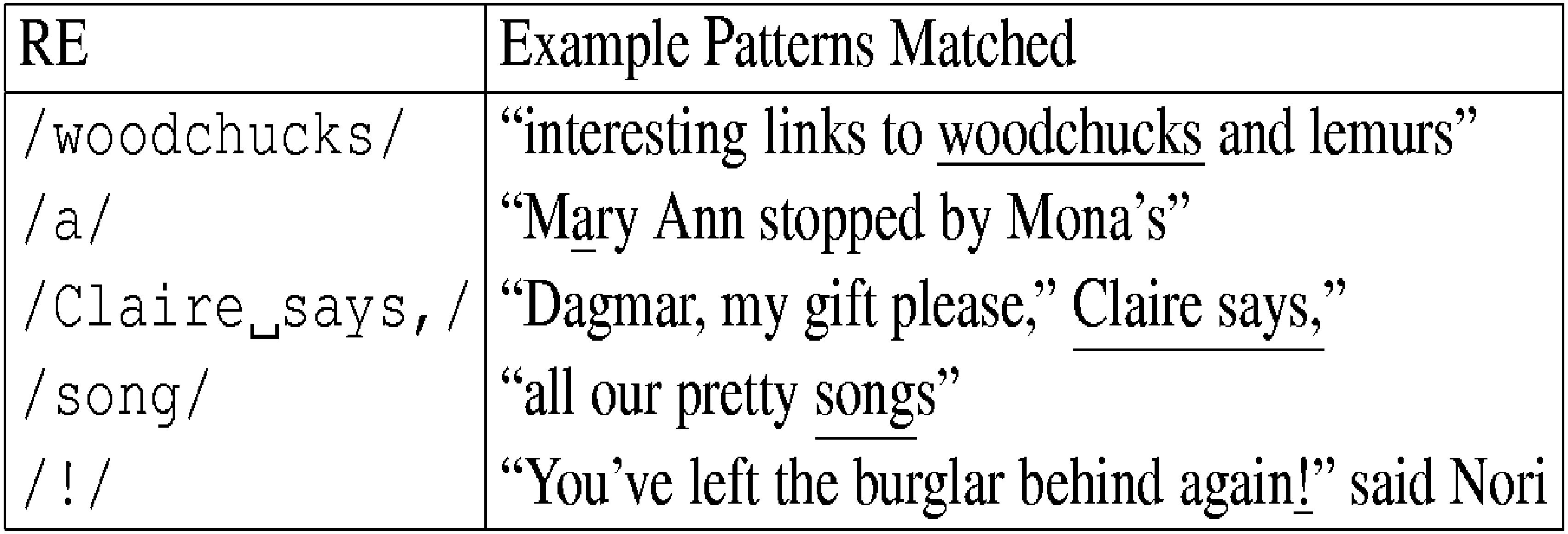
Basic Regular Expression Patterns

- The simplest kind of regular expression is a sequence of simple characters.
  - Example:
    - To search for **woodchuck**, we type /woodchuck/.
  - So the regular expression /Buttercup/ matches any string containing the substring Buttercup,
    - *Example:* I'm called little Buttercup

NOTE:

we will put slashes around each regular expression to make it clear

Regular Expressions

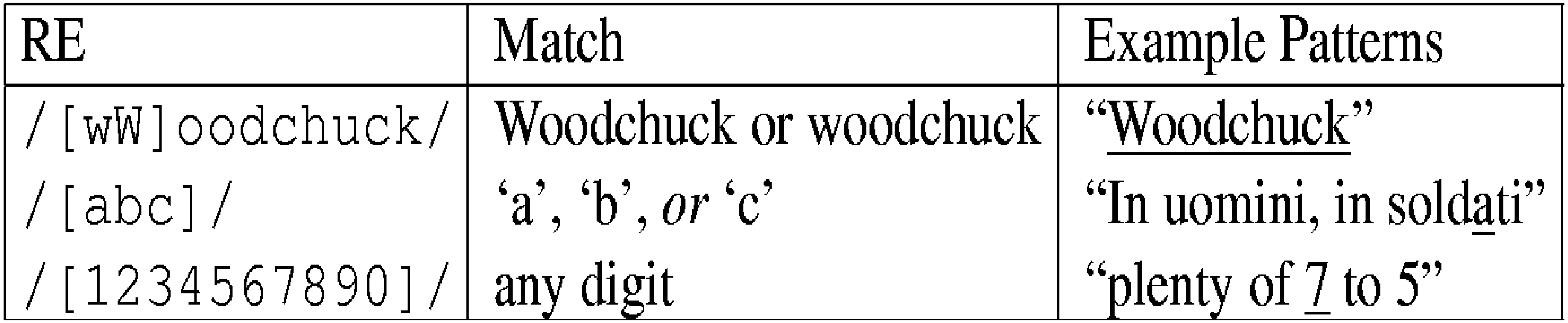


RE	Example Patterns Matched
/woodchucks/	“interesting links to woodchucks and lemurs”
/a/	“Mary Ann stopped by Mona’s”
/Claire␣says,/	“Dagmar, my gift please,” Claire says,”
/song/	“all our pretty songs”
/!/	“You’ve left the burglar behind again!” said Nori

- Regular expressions are case sensitive.

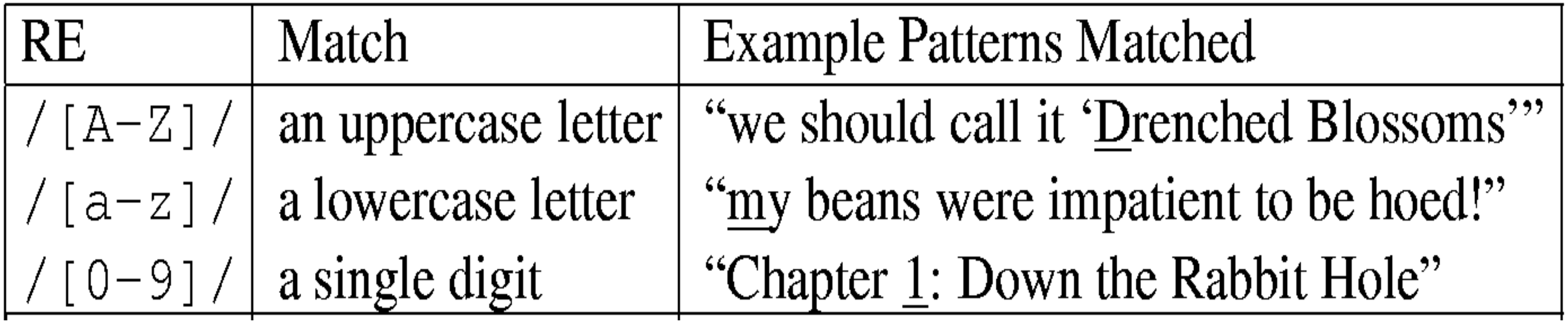
The brackets [ ]

The use of the brackets [ ] to specify a disjunction of characters.



RE	Match	Example Patterns
/[wW]oodchuck/	Woodchuck or woodchuck	“Woodchuck”
/[abc]/	‘a’, ‘b’, *or* ‘c’	“In uomini, in soldati”
/[1234567890]/	any digit	“plenty of 7 to 5”

The use of the brackets [ ] plus the dash - to specify any one character in a range.

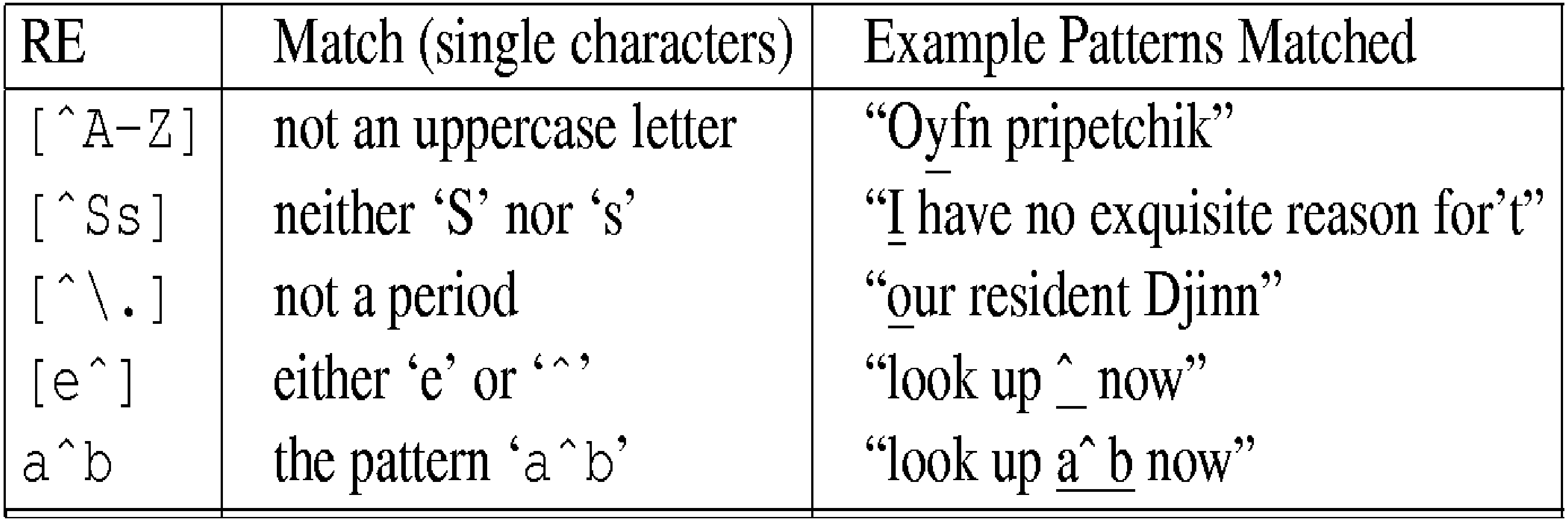


RE	Match	Example Patterns Matched
/[A-Z]/	an uppercase letter	“we should call it ‘Drenched Blossoms’ ”
/[a-z]/	a lowercase letter	“my beans were impatient to be hoed!”
/[0-9]/	a single digit	“Chapter 1: Down the Rabbit Hole”

caret (^)

- The brackets can also be used to specify what a single character cannot be, by use of the caret (^).
- If the caret **^** is the first symbol after the open brackets [, the resulting pattern is **negated**.

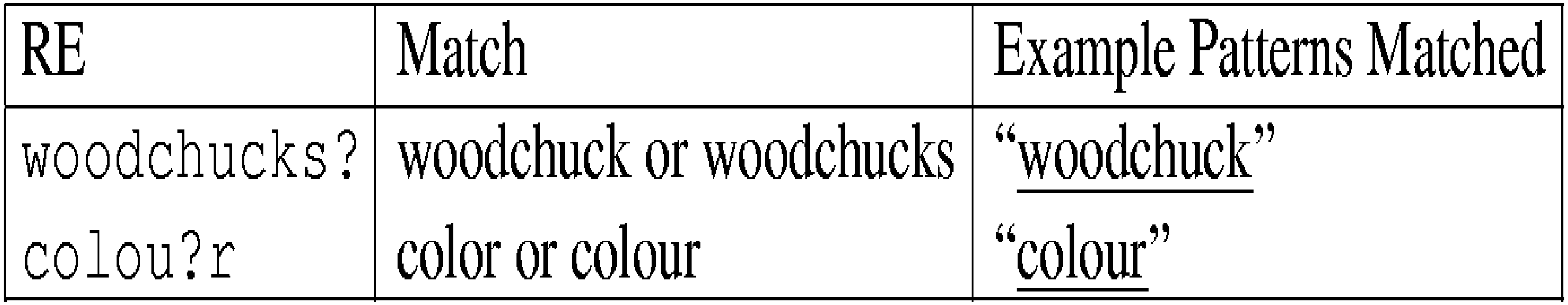
Example: the pattern / [**^a**] / matches any single character (including special characters) except ***a***.



RE	Match (single characters)	Example Patterns Matched
[^A-Z]	not an uppercase letter	“Oyfn pripetchik”
[^Ss]	neither ‘S’ nor ‘s’	“I have no exquisite reason for’t”
[^\.]	not a period	“our resident Djinn”
[e^]	either ‘e’ or ‘^’	“look up ^ now”
a^b	the pattern ‘a^b’	“look up a^ b now”

Question-mark /?/

- Optional elements
- How do we specify both woodchuck and woodchucks?
- The question-mark /?/, means “the preceding character or nothing”.



RE	Match	Example Patterns Matched
`woodchucks?`	woodchuck or woodchucks	“woodchuck”
`colou?r`	color or colour	“colour”

Kleene *

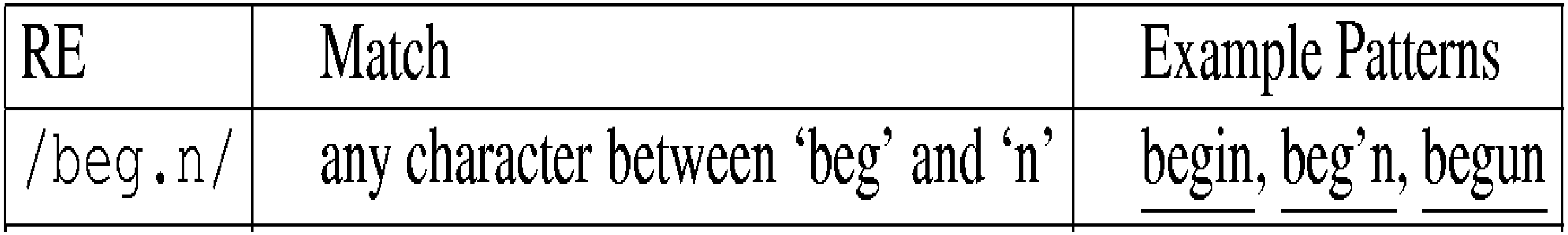
- The Kleene * (pronounced “cleany star”) means 'zero or more occurrences of the immediately previous character or regular expression'.
  - So /a*/ means’ any string of zero or more a's’.
    - This will match a or aaaaaa
  - Write RE for: An integer (a string of digits)?

Kleene +

- This is the Kleene +, which means 'one or more of the previous character'.
- Thus the expression / [0- 9] + / is the normal way to specify 'a sequence of digits'.

The period (/./)

- One very important special character is the period (/./, a wildcard expression that matches any single character (except a carriage return).



RE	Match	Example Patterns
/beg.n/	any character between 'beg' and 'n'	begin, beg'n, begun

Anchors (caret ^)

- **Anchors** are special characters that anchor regular expressions to particular places in a string.
  - The most common anchors are the caret ^ and the dollar-sign $.
  - The caret ^ matches the start of a line.
    - The pattern / ^ The/ matches the word ***The*** only at the start of a line.

Anchors (dollar-sign $)

- The most common anchors are the caret ^ and the dollar-sign $.
  - The caret ˆ matches the **start** of a line
  - The dollar sign **$** matches the **end of a line** (a space at the end of a line)

Ex.: / **^ The dog\. $** / matches a line that contains only the phrase ***The dog***.

- use the backslash here since we want the **.** to mean 'period' .

Anchors (\b & \B)

- There are also two other anchors:
  - **\b** matches a word boundary,
  - **\B** matches a non-boundary.
  - Thus /\bthe\b/ matches the word *the* but not the word *other*.

- Guess , **/\b99/** will match ???

Special characters

- Special characters for **start** and **end**:
- /^man/ => any sequence which begins with "man": *man, manned, manning*...
- /man$/ => any sequence ending with "man": *human, policeman ...*
- /^man$/=> any sequence consisting of "man" only

Disjunction, Grouping, and Precedence

- In such a case, we might want to search for either the string *cat* or the string *dog*.
  - we need a new operator, **the disjunction** operator, also called the pipe symbol **|**.
  - The pattern /cat | dog/ matches either the string cat or the string dog.

Disjunction, Grouping, and Precedence

- For example, How can I specify both guppy and guppies?
  - Cannot say **/ guppy I ies/,** because that would match only the strings *guppy* and *ies*.
  - This is because sequences like guppy take precedence over the disjunction operator **|**.

- So the pattern /gupp (y|ies) / would specify that we meant the disjunction only to apply to the suffixes y and ies.

Quantifiers

- **{m}**
  - Specifies that exactly ***m* copies** of the previous RE should be matched.
    - **Ex.:** a{6} will match exactly six 'a' characters, but not five.

- **{m, n}**
  - Causes the resulting RE to match from *m* to *n* repetitions of the preceding RE.
    - For example, a{3,5} will match from 3 to 5 'a' characters.
  - Omitting *m* specifies a lower bound of zero,
  - Omitting *n* specifies an infinite upper bound.
    - **Ex.:** a{4,} b will match aaaab or a thousand ‘a’ characters followed by a b, but not aaab.

Quantifiers

- /ba*/
  - matches *b, ba, baa, baaa*
  - /*/ means “zero or more of the preceding character or group”
- /(ba){1,3}/
  - matches *ba*, *ba ba* or *ba ba ba*
  - {n, m} means “between n and m”
- /(ba){2}/
  - matches *ba ba*
  - {n} means “exactly n”

Operator Precedence

1. Parentheses	( )
2. Counters	* + ? { }
3. Sequence of Anchors	the ^my end $
4. Disjunction	|

- **Example:**
  - /moo+/
  - /try|ies/
  - /and|or/

Exercise

- Write a regular expression to find all instances of the determiner “*the*”:

  The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

Exercise

- **/the/**

  The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

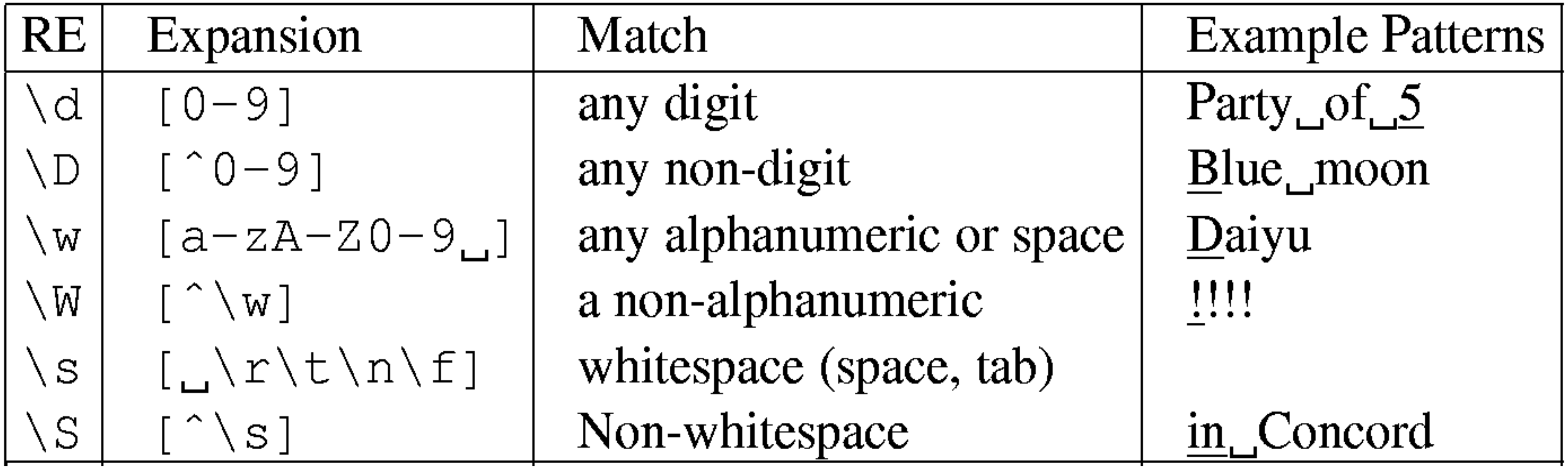
- ***/[Tt]he/***

  The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

- **/\b[Tt]he\b/**

  The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

Advanced Operators

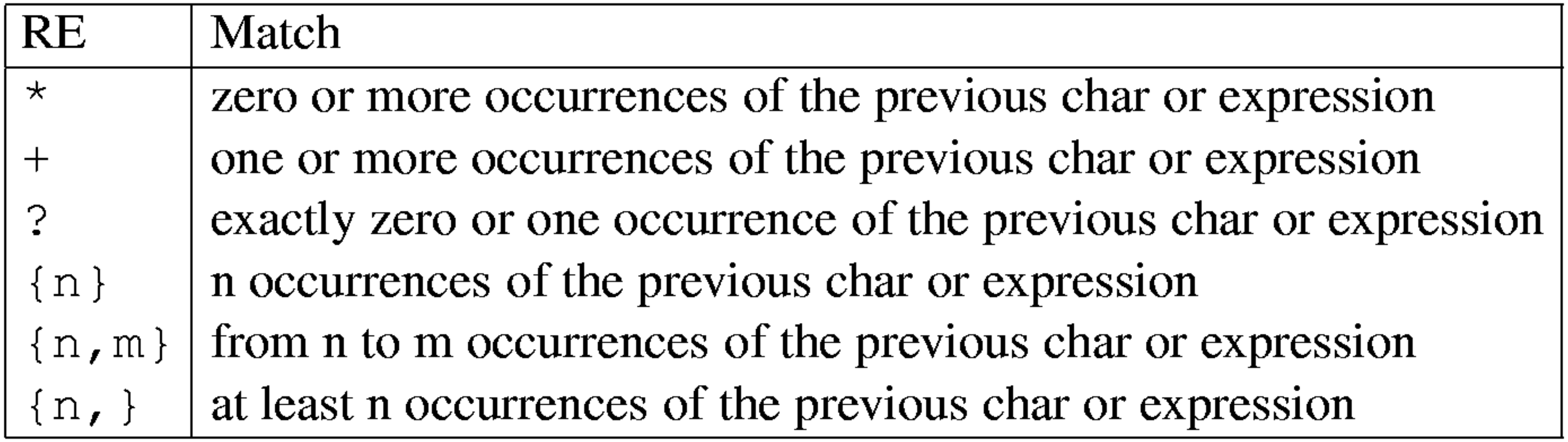


RE	Expansion	Match	Example Patterns
\d	[0-9]	any digit	Party␣of␣5
\D	[^0-9]	any non-digit	Blue␣moon
\w	[a-zA-Z0-9␣]	any alphanumeric or space	Daiyu
\W	[^\w]	a non-alphanumeric	!!!!
\s	[␣\r\t\n\f]	whitespace (space, tab)	
\S	[^\s]	Non-whitespace	in␣Concord

- Finally, certain special characters are referred to by special notation **based on the backslash (/).**
  - the newline character /n and the tab character /t.
- To refer to characters that are special themselves, (like., *, [, and/), precede them with a backslash, (i.e. / \ . /, / * /, / \[/, and / \\ /).

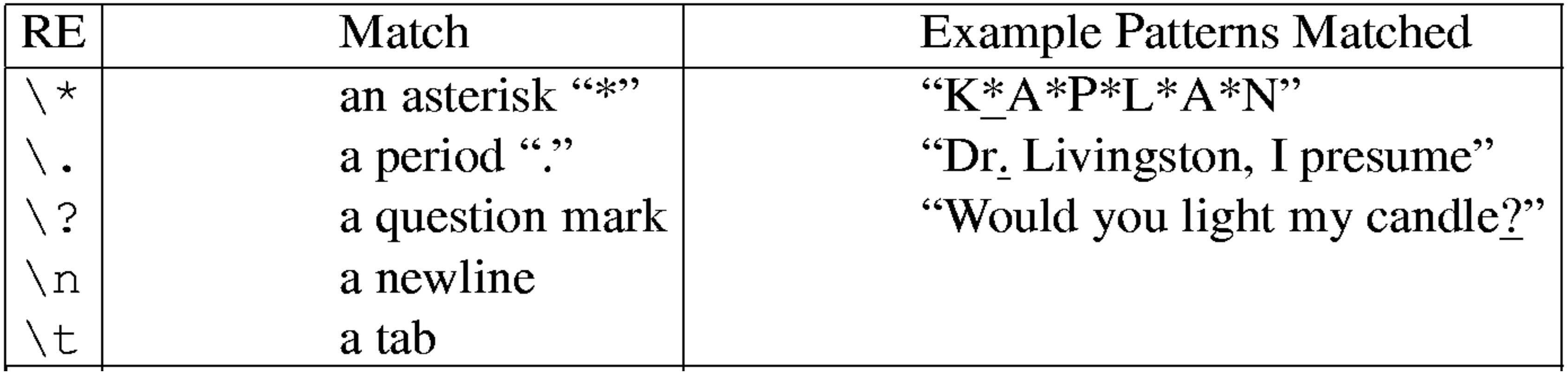
Regular Expression

Regular expression operators for counting



RE	Match
*	zero or more occurrences of the previous char or expression
+	one or more occurrences of the previous char or expression
?	exactly zero or one occurrence of the previous char or expression
{n}	n occurrences of the previous char or expression
{n,m}	from n to m occurrences of the previous char or expression
{n,}	at least n occurrences of the previous char or expression

Some characters that need to be backslashed



RE	Match	Example Patterns Matched
*	an asterisk “*”	“K*A*P*L*A*N”
\.	a period “.”	“Dr. Livingston, I presume”
\?	a question mark	“Would you light my candle?”
\n	a newline	
\t	a tab	

Exercise

- Write RE to represent fractions of dollars. ($199.99, ….)
- Write RE to represent:

  “any computer with more than 6 GHz and 500 GB of disk space for less than $1000”.

Regular Expression Substitution

- An important use of regular expressions is in substitutions.

- We'd like a way to refer back to the integer, we've found so that we can easily add the brackets.
  - To do this, we put parentheses (and) around the first pattern,
  - and use the number operator \1 in the second pattern to refer back. Here's how it looks:
- s/([0-9] +)/<\1>/
- **Ex.:** changing : 35 to <35>

Regular Expression Substitution

- The parenthesis and number operators can also be used to specify that a certain string or expression must occur twice in the text.

i.e. : suppose we are looking for the pattern

'the Xer they were, the Xer they will be

Example :

/the (.*) er they (.*) , the \1er they \2/

The bigger they were, the bigger they were

but not **The bigger they were, the bigger they will be.**

❑ These numbered memories are called registers

Non-capturing Group

- Occasionally we might want to use **parentheses** for grouping, but **don't** want to capture the resulting pattern in a register.
  - use a **non-capturing group**, which is specified by putting the commands group **?:** after the open parenthesis.

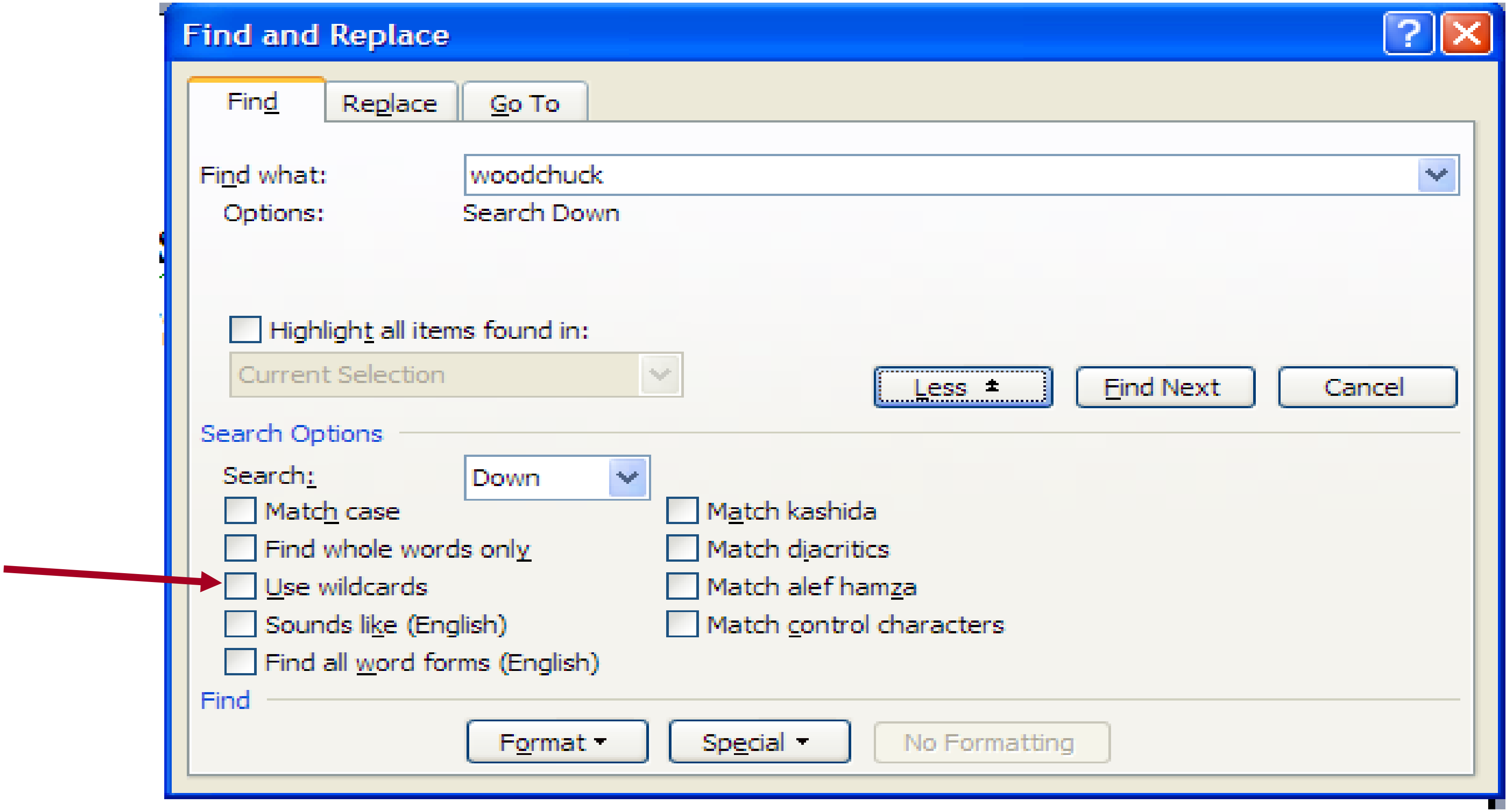
(?: pattern)

- **Ex.:**

/(?:some|a few) (people|cats) like some \1/

Report

- Try regular expressions in MS WORD in both Arabic & English



Find and Replace

Find what: woodchuck

Options: Search Down

□ Highlight all items found in:

Current Selection

Search Options

Search: Down

□ Match case

□ Find whole words only

□ Use wildcards

□ Sounds like (English)

□ Find all word forms (English)

□ Match kashida

□ Match diacritics

□ Match alef hamza

□ Match control characters

- Write RE can match "gray" or "grey"
  - gray|grey
  - gr(a|e)y

- a|b* and (a|b)* are equivalent
  - a|b* denotes {ε, "a", "b", "bb", "bbb", ...}
  - (a|b)* denotes the set of all strings with no symbols other than "a" and "b", including the empty string: {ε, "a", "b", "aa", "ab", "ba", "bb", "aaa", ...}

- Normalizing text means converting it to a more convenient, standard form
- Another part of text normalization is lemmatization, the task of determining that two words have the same root, despite their surface differences
- Stemming refers to a simpler version of lemmatization in which we mainly just strip suffixes from the end of the word.