

# **REGULAR EXPRESSIONS (APPENDIX A)**

# BASIC IDEA

- regular expression: defines a set of one or more strings of characters
- simple string of characters -> represents itself
- special/metacharacters -> characters that do not represent themselves
- add in special characters -> match a pattern which can represent many strings

# SPECIAL CHARACTERS

- We'll focus on ERE (extended regular expression syntax)
- delimiter:
  - marks beginning/end of expression,
  - often /
  - some utilities let you use other delimiters

Char	Use	Example
\	escape special character	a\+b matches "a+b"
.	wildcard - match any character	.ord matches "word", "cord"
[ ]	character class	[bB]ob matches "bob", "Bob"
^	beginning of line	^B matches "B" at start of line
\$	end of line	!\$ matches "!" ending line



Char	Use	Example
*	match 0 or more occurrences of preceding	<code>bo*</code> matches "b", "boooooo"
?	match 0 or 1 occurrences of preceding	<code>bo?</code> matches "b", "bo"
+	match 1 or more occurrences of preceding	<code>bo+</code> matches "bo", "booooo"
( )	group characters	<code>(da)*</code> matches "da", "dada"
	match previous or next	<code>hi bye</code> match "hi", "bye"



## RECALL: GREP

- We used to search for lines containing certain string
- grep: global regular expression print
- Can search for patterns, not just specific strings by expressing regex
- Add `-E` to search for ERE pattern



# MISC NOTES

- Matches longest possible string
- Can also use character classes (like with `tr`)
- Examples:
  - `[[:alpha:]]`
  - `[[:alnum:]]`
  - `[abc]`
  - `[^a-z]` everything except a through z