1 – What are Regular Expressions?

Week 14 Presentation 1

Regular Expressions

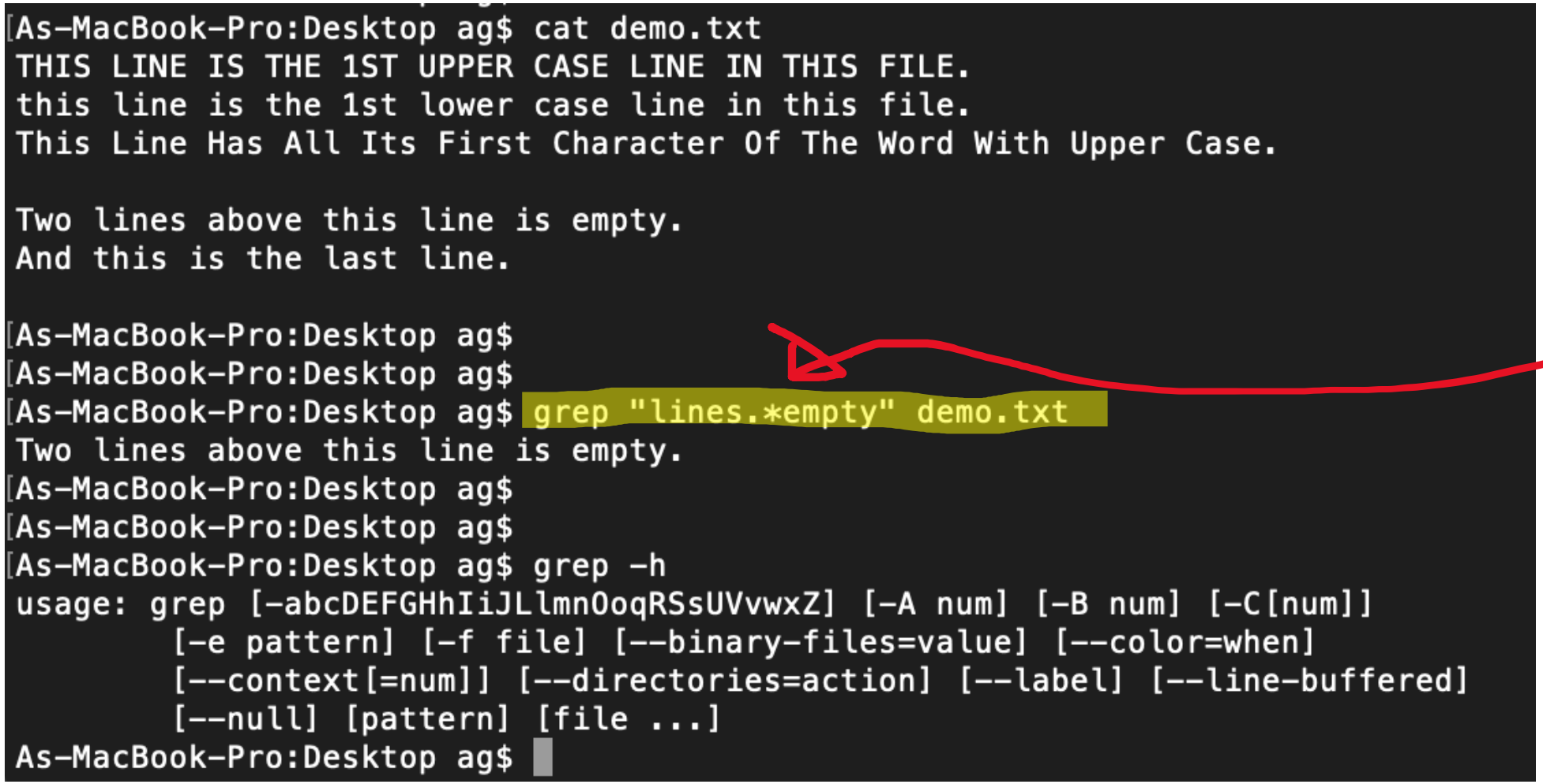
- Regular expression (also called regex or RE) is useful for describing search patterns for matching text.
- A Regex is just a sequence of some characters that defines a search pattern.
- Regex are applied widely for parsing, filtering, validating, and extracting meaningful information from large text, such as logs and output generated from other programs.

Some Background

- The term *regular expression comes from mathematics and computer science theory,* where it reflects a trait of mathematical expressions called *regularity. Such an expression* can be implemented in software using a deterministic finite automaton (DFA). A DFA is a finite state machine that doesn't use backtracking.
- The text patterns used by the earliest *grep tools were regular expressions in the mathematical* sense. Though the name has stuck, modern-day Perl-style regular expressions are not regular expressions at all in the mathematical sense.
- They're implemented with a nondeterministic finite automaton (NFA). You will learn all about backtracking shortly. All a practical programmer needs to remember from this note is that some ivory tower computer scientists get upset about their well-defined terminology being overloaded with technology that's far more useful in the real world.

GREP

file



Regex: grep returns lines matching the regex pattern

Formal Languages

Some definitions…

- An alphabet is a set of symbols
- A string is a finite sequence of alphabet symbols
- A formal language is a set of strings (possibly infinite) over the same alphabet

formal language	*in the language*	*not in the language*
second-to-last symbol is a	aa bbbab bbbbbbbbbababab	a aaaba bbbbbbbbbbbbb
equal numbers of as and bs	ba bbaaba aaaabbbbbbbaaaba	a bbbaa abababababababa
palindromes	a aba abaabaabaaba	ab bbbba abababababababab
contain the pattern abba	abba abaababbabbababbba bbbbbbbbbbabbabbbbb	abb bbabaab aaaaaaaaaaaaaaaaa
number of bs is divisible by 3	bbb baaaaabaaaab bbbabbaaabaaababababaaa	bb abababab aaaaaaaaaaaaaaaab

Examples of formal languages over a binary alphabet

Binary Strings involving 2 symbols (here {a,b})

More Examples of Formal Languages

	symbols	*symbol name*	*string name*
binary	01 (or ab)	bit	bitstring
Roman	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ	letter	word
decimal	0123456789	digit	integer
special	~`!@#$%^&*()_-+={[}]|\:;"'<,>.?/		
keyboard	*Roman* + *decimal* + *special*	keystroke	typescript
genetic code	ATCG	nucleotide base	DNA
protein code	ACDEFGHIKLMNPQRSTVWY	amino acid	protein
ASCII	see SECTION 6.1	byte	`String`
Unicode	see SECTION 6.1	char	`String`

Commonly used alphabets and associated terminology

More Examples of Formal Languages

formal language	*in the language*	*not in the language*
amino acid encodings	AAA AAC AAG AAT ACA ACC ACG ACT TAC TAT TGC TGG TGT	TAA TAG TGA AAAAAAAAA ABCDE
U.S. telephone number	(609) 258-3000 (800) 555-1212	(99) 12-12-12 2147483648
English words	and middle computability	abc niether misunderestimate
legal English sentences	This is a sentence. I think I can.	xya a b.c.e?? Cogito ergo sum.
legal Java identifiers	a class $xyz3_XYZ	12 123a a((BC))*
legal Java programs	public class Hi { public static void main(String[] args) { } }	int main(void) { return 0; }

Regular expressions. A *regular expression (RE)* is a string of symbols that specifies a formal language. We define what regular expressions are (and what they mean) recursively, using the union, concatenation, and closure operations on sets, along with parentheses for operator precedence. Specifically, every regular expression is either an *alphabet symbol*, specifying the singleton set containing that symbol, or composed from the following operations (where `R` and `S` are REs):

- *Union:* `R | S`, specifying the union of the sets `R` and `S`,
- *Concatenation:* `RS`, specifying the concatenation of the sets `R` and `S`,
- *Closure:* `R*`, specifying the closure of the set `R`,
- *Parentheses:* `(R)`, specifying the same set as `R`.

Definition. A language is *regular* if and only if it can be specified by an RE.

Using Regular Expressions to Make Many Text Processing Problems Solvable Quickly and Easily

- You might wonder why we would ever need to use regular expressions so why learn about them?
- Here are some use cases:
  - Searching for text where we don’t know the specific text we are looking for up front but we do know some rules or patterns in the text:
    - Search for an IP or MAC address in a log (e.g. web server access log)
    - Search for a 10 digit mobile number that may optionally be preceeded with a country code
  - Searching where the length of the text to extract is not known beforehand:
    - Search for URLs that start with **http://** or **https://**

Using Regular Expressions to Make Many Text Processing Problems Solvable Quickly and Easily

- Generating tokens by splitting a given text on delimiters of a variable type and length
- Extracting text that lies between 2 or more search patterns
- Validating input from the user (account number, usernames, credit card number etc)
- Getting parts of a text with some properties such as repeated words
- Conversions to custom predefined formats: e.g. insert comma after every three digits in numbers or remove commas that occur in parantheses
- Doing a global search replace while skipping escaped characters

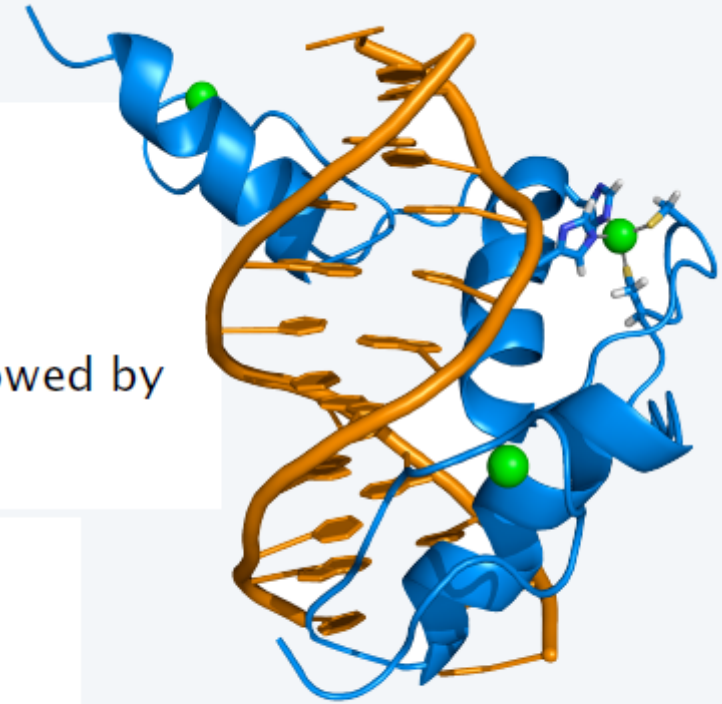
Pattern matching

Pattern matching problem. Is a given string an element of a given set of strings?

Example 1 (from computational biochemistry)

An amino acid is represented by one of the characters C A V L I M C R K H D E N Q S T Y F W P.

A protein is a string of amino acids.



A C_2H_2-type zinc finger domain signature is

- C followed by 2, 3, or 4 amino acids, followed by
- C followed by 3 amino acids, followed by
- L, I, V, M, F, Y, W, C, or X followed by 8 amino acids, followed by
- H followed by 3, 4, or 5 amino acids, followed by H.

Q. Is this protein in the C_2H_2-type zinc finger domain?

A. Yes.

Pattern matching

Example 2 (from commercial computing)

An e-mail address is

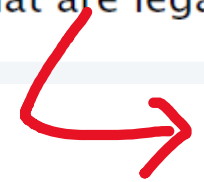
- A sequence of letters, followed by
- the character "@", followed by
- followed by a nonempty sequence of lowercase letters, followed by the character "."
- [any number of occurrences of the previous pattern]
- "edu" or "com" (others omitted for brevity).

Q. Which of the following are e-mail addresses?

	A.
rs@cs.princeton.edu	✓
not an e-mail address	✗
wayne@cs.princeton.edu	✓
eve@airport	✗
rs123@princeton.edu	✗

Oops, need to fix description →

Challenge. Develop a precise description of the set of strings that are legal e-mail addresses.



search online

Pattern matching

Example 3 (from genomics)

A nucleic acid is represented by one of the letters a, c, t, or g.

A genome is a string of nucleic acids.

A Fragile X Syndrome pattern is a genome having an occurrence of gcg, followed by any number of cgg or agg triplets, followed by ctg.

Note. The number of triplets correlates with Fragile X Syndrome, a common cause of mental retardation.

Q. Does this genome contain a such a pattern?

gcggcgtgtgtgcgagagagtgggtttaaagctg**gcg**|**cgg**|**agg**|**cgg**|**ctg**gcgcggaggctg

A. Yes.