

# applied regex

---



C

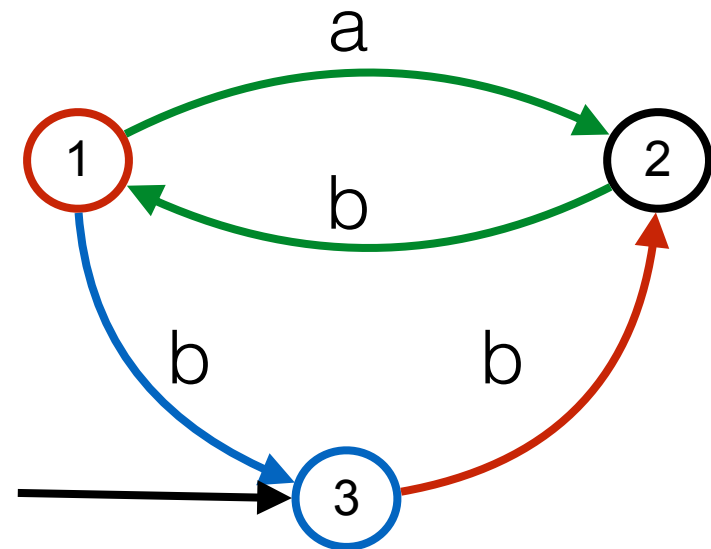
- ✓ implementing REs using finite state automata
- using REs to find patterns

Is there a regular expression for every  
FSM?

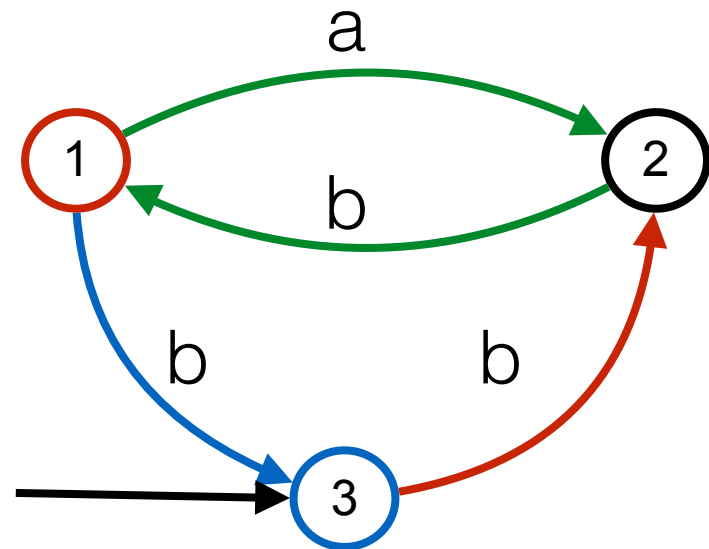


$$L_3 = (b \ b \ (a \ b)^* \ b)^*$$

$$L_3 = (b \ (b \ a)^* \ b \ b)^*$$



Is there a regular expression for every  
FSM?



$$L_3 = (b\ b\ (a\ b)^*\ b)^*$$

$$L_3 = (b\ (b\ a)^*\ b\ b)^*$$

$$L_3 = \varepsilon \mid b\ (b\ b\ b \mid b\ a)^*\ b\ b$$

# Arden's Lemma

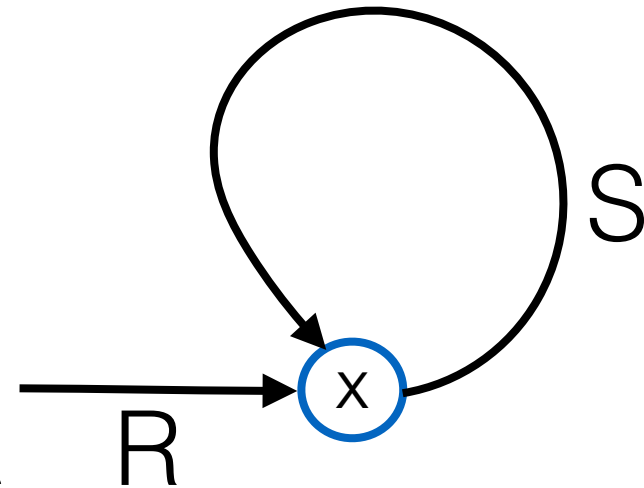


If  $R$  and  $S$  are regular expressions then the equation

$$X = R \mid X S$$

has a solution  $X = R S^*$

If  $\epsilon \notin L(S)$  then this solution is unique.



# REs and FSAs

- Regular expressions can be viewed as a textual way of specifying the structure of finite-state automata
- Finite-state automata are a way of implementing regular expressions

# Regular expressions

- A formal language for specifying text strings
- How can we search for any of these?

- ◆ woodchuck
- ◆ woodchucks
- ◆ Woodchuck
- ◆ Woodchucks



# Regular Expressions for Textual Searches

Who does it?

Everybody:

- Web search engines, CGI scripts
- Information retrieval
- Word processing (Emacs, vi, MSWord)
- Linux tools (sed, awk, grep)
- Computation of frequencies from corpora
- Perl

# Regular Expression

- **Regular expression:** formula in algebraic notation for specifying a set of strings
- **String:** any sequence of alphanumeric characters
  - letters, numbers, spaces, tabs, punctuation marks
- **Regular expression search**
  - **pattern:** specifying the set of strings we want
  - **corpus:** the texts we want to search through

<http://www.inf.ed.ac.uk/teaching/courses/inf1/cl/tools/regex-crib.xml>



[PDF] Bulk data transfer toolkit: validation rules for CAS ... - Gov...

[https://www.gov.uk/.../4\\_\\_Bulk\\_Data\\_Transfer\\_-\\_additional\\_validation\\_...](https://www.gov.uk/.../4__Bulk_Data_Transfer_-_additional_validation_...) ▼

If the country code given for the address is 'GBR' then the field is validated against the UK postcode **regular expression**. (see section 3) If the address is given as ...

You visited this page on 14/11/14.

## 3. UK postcode regular expression

---

The following is the UK Postcode Regular Expression and the corresponding detail explaining the logic behind the UK Postcode Regular Expression.

### 3.1 Expression

```
^([Gg][Ii][Rr] 0[Aa]{2})|((([A-Za-z][0-9]{1,2})|(([A-Za-z][A-Ha-hJ-Yj-y][0-9]{1,2})|((([AZa-z][0-9][A-Za-z])|([A-Za-z][A-Ha-hJ-Yj-y][0-9]?[A-Za-z]))) [0-9][A-Za-z]{2}))$
```

## 3.2 Logic

"GIR 0AA"

OR

One letter followed by either one or two numbers

OR

One letter followed by a second letter that must be one of ABCDEFGHJ  
KLMNOPQRSTUVWXYZ (i.e..not I) and then followed by either one or two  
numbers

OR

One letter followed by one number and then another letter

OR

A two part post code

where the first part must be

One letter followed by a second letter that must be one of ABCDEFGH  
JKLMNOPQRSTUVWXYZ (i.e..not I) and then followed by one number and  
optionally a further letter after that

AND

The second part (separated by a space from the first part) must be One  
number followed by two letters.

A combination of upper and lower case characters is allowed.

**Note:** the length is determined by the regular expression and is between 2 and 8 characters.

## Browse Expressions by Category



Email

Uri

Numbers

Strings

Dates and Times

Misc

Address/Phone

Markup/Code

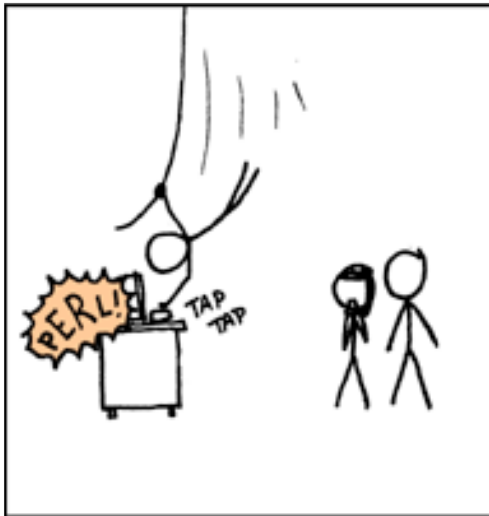
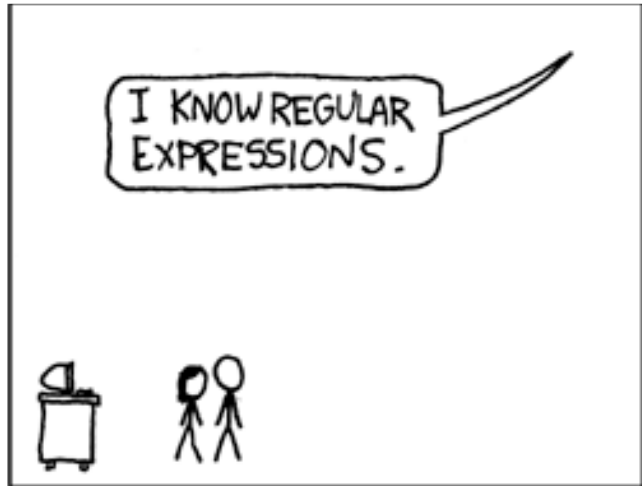
Welcome to RegExLib.com, the Internet's first Regular Expression Library. Currently we have indexed 4736 expressions from 2214 contributors around the world.

<http://www.regexlib.net/>

WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.



<http://xkcd.com/>



# Everyman crossword No 3,551

[Print version](#) | [Blind & PS version](#) | [PDF version](#)

---

The Observer, Sunday 26 October 2014 00.00 GMT

---



**3** Typesetter in awfully poor sitcom (10)

```
% cat /usr/share/dict/words | egrep ^[poorsitcom]{10}$
```

```
$ cat /usr/share/dict/words | egrep ^[poorsitcom]{10}$
```

compositor  
copromisor  
crisscross  
isoosmosis  
isotropism  
microtomic  
optimistic  
poroscopic  
postcosmic  
postscript  
prioristic  
promitosis  
proproctor  
protoprism  
tricrotism  
troostitic

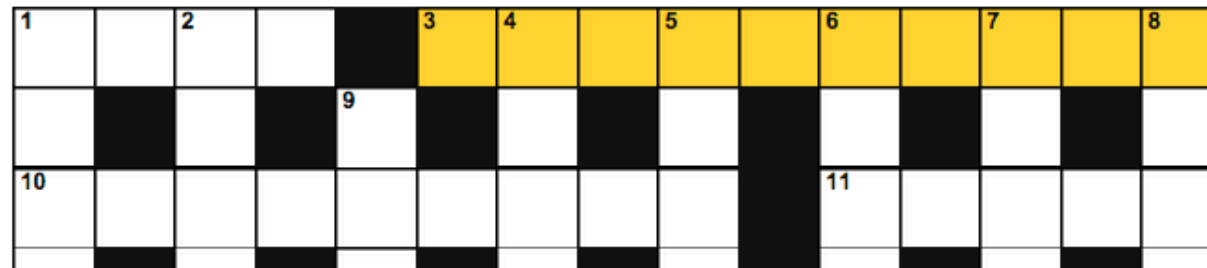
## Everyman crossword No 3,551

[Print version](#) | [Blind & PS version](#) | [PDF version](#)

---

The Observer, Sunday 26 October 2014 00.00 GMT

---



**3** Typesetter in  
awfully poor sitcom  
(10)

```
% cat /usr/share/dict/words | egrep ^[poorsitcom]{10}$ | grep o.*o.*o
```

compositor  
copromisor  
isoosmosis  
poroscopic  
proproctor

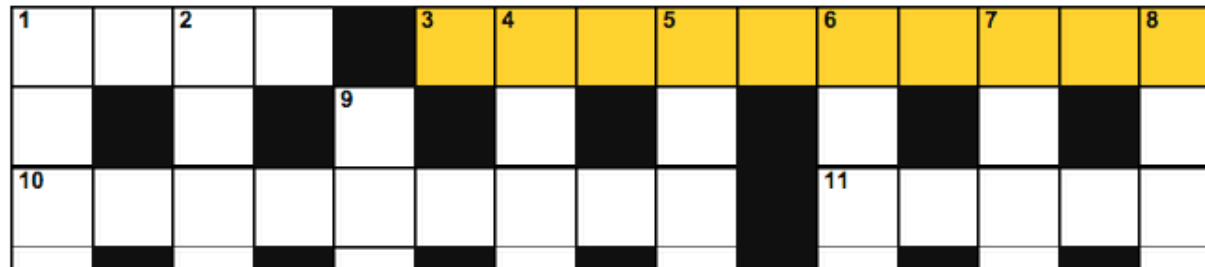
## Everyman crossword No 3,551

[Print version](#) | [Blind & PS version](#) | [PDF version](#)

---

The Observer, Sunday 26 October 2014 00.00 GMT

---



**3** Typesetter in  
awfully poor sitcom  
(10)

# The Turing Test



- Aim is to answer the question

Q1: “Can a machine think?”

- But what do we mean by “think”? and “machine”?

- Turing chose to replace Q1 by Q2

Q2: “Are there imaginable digital computers which would do well in the [Turing Test]?”

Q3: “Can machines do what we (as thinking entities) can do?”

- Turing argues Q3 draws "a fairly sharp line between the physical and intellectual capacities of a man."



# The Turing Test: Background

- Participating in a natural language conversation is hard:
  - ▶ ambiguity
  - ▶ vagueness
  - ▶ context
  - ▶ real world knowledge
  - ▶ .....
- Language use is intertwined with general human cognitive abilities
- Ability to process language as people do will signal the arrival of truly intelligent machines

# The Imitation Game and the Turing Test

- Party game in which a man (A) and a woman (B) go into separate rooms, and guests (or an interrogator) (C) try to tell them apart by writing a series of questions and reading the typewritten answers sent back
- Both the man (A) and the woman (B) aim to convince the guests (C) that they are the other
- Replace A or B by a computer
- If this does not significantly change the interrogator's success rate, the computer would have to be judged intelligent

# Turing Test



- A person Q is connected to two others, A and B, through computer networking
- Q interacts verbally (by typing) with A and B
- One of A/B is another human, the other is a computer program
- A and B will both try to convince Q that they are the human
- The test is whether Q can correctly detect which is the computer program
- If Q cannot, the program has passed the Turing test.

# Imitating Communication: Eliza

- Joseph Weizenbaum (1964): fancy linguistic analysis not needed to give the appearance of intelligent conversation
  - simple pattern matching produces responses to typed input
  - people have been fooled by Eliza and similar programs
  - program is very simple but surprisingly deceptive.
- Example: dialogue that supposedly took place between a sales executive who wanted to use the computer and Eliza

# Sample Dialogue

SE: Do you think that I can use the computer this morning?

***El: Why do you ask?***

SE: I might be able to make some additional telecomp sales.

***El: Why aren't you sure?***

SE: My prospective customers have never seen a demonstration of the system.

***El: Does this mean very much to you?***

SE: Of course it does.

***El: You seem very positive about that.***

SE: This is ridiculous.

***El: What do you mean this is ridiculous.***

SE: Is there any reason why I can't use the computer now?

***El: Tell me more about your prospective customers***

SE: Please dial me up on 491-1850

# Eliza fools the SE

- Supposedly, the SE telephoned the operator he thought he had been talking to, and the exchange went something like this:

SE: Why are you being so snotty to me?

Op: What do you mean, why am I being so snotty to you?

*rest censored...*

- Sales Executive (SE) believed he was talking to a computer operator!
- Did not realise that he was, in fact, on-line to Eliza
- Seems that Eliza could fool people into believing that they were talking to another person

# Imitating Communication: Eliza

- Weizenbaum very carefully choose the domain so Eliza could demonstrate its power
- Eliza imitates a Rogerian Psychotherapist
  - ▶ clinician draws patients out to get them to talk about themselves and their problems, without being directive.
- Eliza uses simple pattern matching to carry on limited form of conversation.
- Seems to “Pass the Turing Test!” (*McCorduck, 1979, pp. 225-226*)
- Eliza Demo:

[http://www.lpa.co.uk/pws\\_dem4.htm](http://www.lpa.co.uk/pws_dem4.htm)

# Eliza: Weizenbaum (1966)

User: You are like my father in some ways.

ELIZA: What resemblance do you see?

User: You are not very aggressive but I think you don't want me to notice that.

ELIZA: What makes you think I am not aggressive?

User: You don't argue with me

ELIZA: Why do you think I don't argue with you?

User: You are afraid of me

ELIZA: Does it please you to believe I am afraid of you?



# How Eliza works

- Simple pattern matching with if-then rules:
  - ▶ stores a set of “key words” and “key phrases”
  - ▶ tries to match these to input sentence
  - ▶ if there’s a match, perform simple transformation on sentence to provide response

- Example

Rule: IF input = “I am X”

THEN output = “How long have you been X?”

input: “I am sick”

matches: “I am X” where X = “sick”

response: *How long have you been sick?*

- Large set of if-then (a.k.a. stimulus-response) rules used

# How Eliza works

Set stimulus = input

output “ ”

Until match is found or no more rules in list

IF stimulus matches if-part of rule

THEN substitute values from stimulus for variables in then-part of rule, and  
set output = filled-in pattern

IF output = “ ”, then output = *“Tell me more.”*

Print output

Sample Rule: IF input = “I am X”

THEN output = “How long have you been X?”

# Cascade of substitutions

## Example:

Input: I am sick of you

Rule: IF input = "I am X"

THEN output = "How long have you been X?"

Output: *How long have you been sick of you?*

## ***But this isn't what we want!***

So, Eliza first changes all instances of 2nd person singular pronouns to 1st person singular pronouns, and 1st person singular pronouns to 2nd person singular pronouns, e.g.,

you --> me

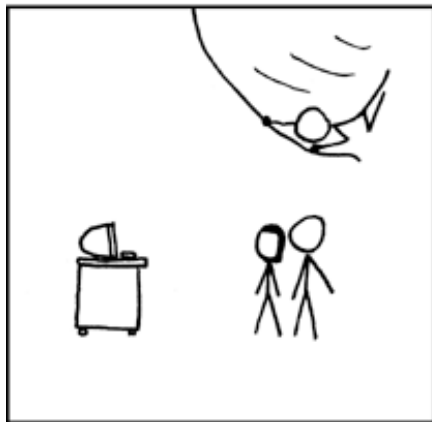
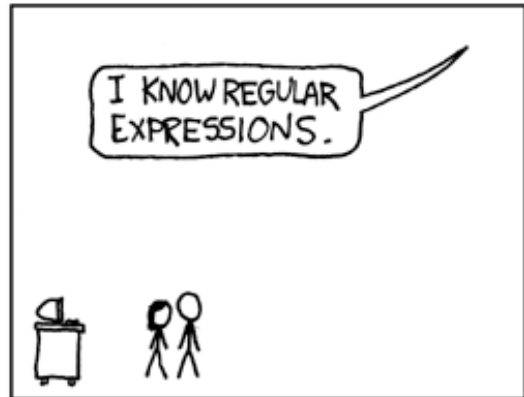
my --> your

I'm --> You are

Output: *How long have you been sick of me?*

# You could write Eliza!

WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.



# Regular Expressions

- Basic regular expression patterns
- Java-based syntax
- **Disjunctions** `[mM]`

Reg Exp	Match	Example Patterns
<code>[mM]other</code>	mother or Mother	“ <b>M</b> other”
<code>[abc]</code>	a or b or c	“you <b>a</b> re”
<code>[1234567890]</code>	any digit	“ <b>3</b> times a day”

# Regular Expressions

- **Ranges** [A-Z]

RE	Match	Examples Patterns Matched
[A-Z]	an uppercase letter	“call me <b>E</b> liza”
[a-z]	a lowercase letter	“ <b>c</b> all me Eliza”
[0-9]	a single digit	“I’m off at <b>7</b> ”

- **Negations** [^Ss]

RE	Match	Examples Patterns Matched
[^A-Z]	not an uppercase letter	“ <b>Y</b> ou can call me Eliza”
[^Ss]	neither s nor S	“ <b>S</b> ay hello Eliza”
[^\.]	not a period	“ <b>H</b> ello.”

# Regular Expressions

- **Optional characters: ? , \* and +**

- ? (0 or 1)

colou?r → color or colour

- \* (0 or more)

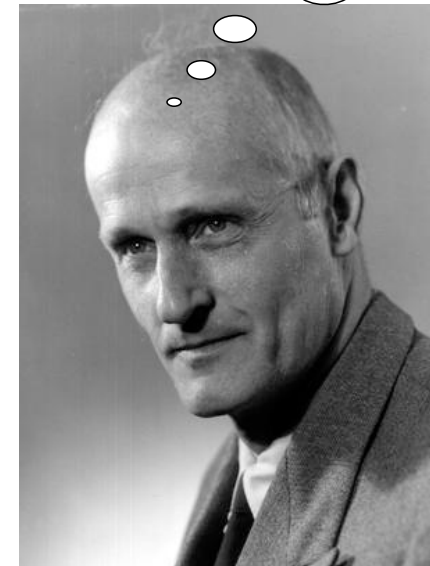
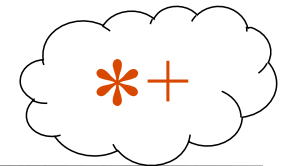
oo\*h! → oh! or ooh! or ooooh!

- + (1 or more)

o+h! → oh! or ooh! or ooooh!

- . any char except newline

beg.n → begin or began or begun



*Stephen Cole Kleene*

# Regular Expressions

- **Anchors `^` and `$`**

- `^[A-Z]` → "France", "Paris"
- `^[^A-Z]` → "¿verdad?", "really?"
- `\.$` → "It's over ."
- `moo$` → "moo", but not "mood"

- **Boundaries `\b` and `\B`**

- `\bon\b` → "on my way" "Monday"
- `\Bon\b` → "automaton"

- **Disjunction `|`**

- `yours|mine` → "it's either yours or mine"



# Regular Expressions

---

<http://www.inf.ed.ac.uk/teaching/courses/il1/2010/labs/2010-10-28/regexrepl.xml>

- **Replacement**
  - in emacs
  - in javascript
  - in python and perl
  - ...

`s/\bI ('m| am) \b /ARE YOU/g`

- Syntax varies - the ideas are universal

# Back to Eliza

---

User: Men are all alike

ELIZA: IN WHAT WAY

User: They're always bugging us about something or other

ELIZA: CAN YOU THINK OF A SPECIFIC EXAMPLE?

User: Well, my boyfriend made me come here

ELIZA: YOUR BOYFRIEND MADE YOU COME HERE

User: He says I'm depressed much of the time

ELIZA: DO YOU THINK YOU ARE DEPRESSED MUCH  
OF THE TIME

# Eliza-style regular expressions

Step 1: replace first person with second person references

```
s/\bI ('m| am) \b /ARE YOU/g  
s/\bmy\b /YOUR/g  
S/\bmine\b /YOURS/g
```

Step 2: use additional regular expressions to generate replies

```
s/. * ARE YOU (SAD|DEPRESSED) .* /I AM SORRY TO HEAR YOU ARE $1/  
s/. * ARE YOU (.*) .* /WHY ARE YOU $1/  
s/. * all .* /IN WHAT WAY/  
s/. * always .* /CAN YOU THINK OF A SPECIFIC EXAMPLE/
```

Step 3: use scores to rank possible transformations

# Criticisms of Eliza

- Lack of memory: there is no relation between the current response and any previous stimulus or response
- The program is inflexible
- There is no knowledge of the structure or meaning of the input
  - ▶ garbage in, garbage out!
- The program has no world knowledge.
- Program uses the technique of keyphrase-matching:
  - ▶ an example of pattern-matching (very important)

# Experiment

---

<http://www.inf.ed.ac.uk/teaching/courses/il1/2010/labs/2010-10-28/regexrepl.xml>

- **Replacement**
  - in emacs
  - in javascript
  - in python and perl
  - ...

`s/\bI ('m| am) \b /ARE YOU/g`

- Syntax varies - the ideas are universal