# Module-II (Part-1): Word Level Analysis

by:
***Dr. Soumya Priyadarsini Panda***
Sr. Assistant Professor
Dept. of CSE, SIT, Bhubaneswar

# Use of Regular Expression
# &
# Finite State Automata
# in NLP Application Development

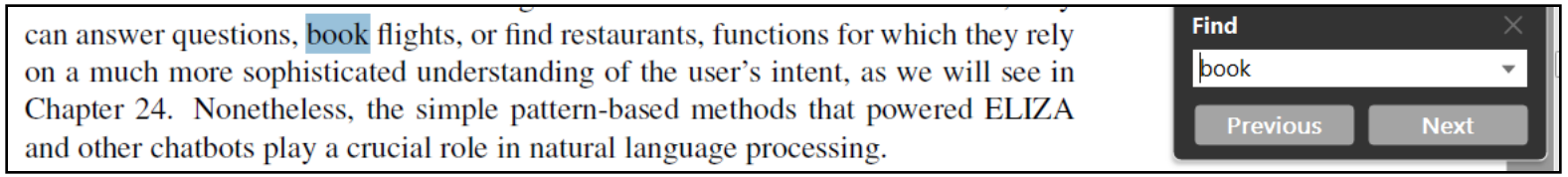
# Regular Expression (RE)

- It is a pattern-matching standard for string parsing and replacement.
- It is used in every computer language, word processor, and text processing tools.
- RE is an algebraic notation for characterizing a set of strings.
- Usage of regular expressions to search for specific text is a useful technique in NLP.
- It is being used in applications like: Text Analysis, Information Extraction, Speech Recognition, etc.

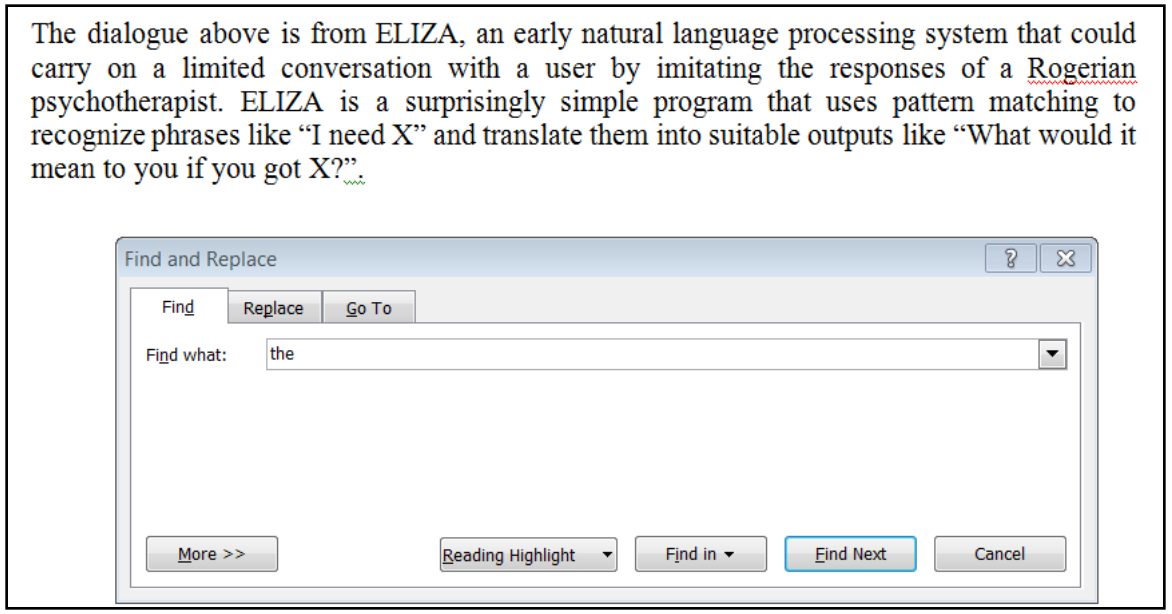
# Example Applications

- Want to search for a sting in a document (ctrl+ F)

- **PDF reader:**




- **MS Word:**



# Example Applications Cont…

- Regular Expressions are used in various tasks such as:
  - data pre-processing
  - rule-based information mining systems
  - pattern matching
  - text feature engineering
  - web scraping
  - data extraction

# Example Applications Cont…

- While working with text data, on building NLP models for:
  - text classification
  - machine translation
  - text summarization, etc
    - A variety of text that comes from diverse sources are required to be processed.
    - The text may contain noisy contents which must be cleaned to make the text data usable for the NLP models.
- RE is used to clean the data (filter out letters, digits, punctuations, symbols, etc)

# Example Regular Expression Patterns

# Example

- Search for a string ‘book’ in text:

RE: / book /

can answer questions, book flights, or find restaurants, functions for which they rely on a much more sophisticated understanding of the user’s intent, as we will see in Chapter 24. Nonetheless, the simple pattern-based methods that powered ELIZA and other chatbots play a crucial role in natural language processing.

- Search for the number 24

RE: / 24 /

- RE to find all occurrences of the character ‘a’

RE: / a /

Sample Text:

“Regular expressions are used in string searching.

# Example Cont…

The RE-

/ book /

: Find all occurrences of the word ‘book’

but not ‘Book’

***Regular expressions are case sensitive***

Example Text:

Book me a ticket. I have not booked yet.

I was busy in reading a book.

RE to find all occurrences of Book/book:

Book | book

or

/ [Bb]ook /

# Basic Regular Expression Patterns

## Character classes

| RE | Details |
|---|---|
| /book/ | Search for the string ‘book’ |
| [abc] | Match any of a, b, or c |
| [a-z] | Match any lower case character |
| [A-Z] | Match any upper case character |
| [^A-Z] | Match any character other than an uppercase letter |
| [^abc] | Match any character other than a, b, c |
| [+*?.] | Match any of +, *,? or . |
| [a^] | Match a or ^ |

# Cont…

| RE | Details |
|---|---|
| [0-9] | Matches any single digit number |
| [0-9]+ | Matches multidigit number |
| [Bb]ook | Matches the string ‘Book’ or ‘book’ |
| Book \| book | Disjunction: Matches the string ‘Book’ or ‘book’ |
| /book**s**?/ | Optional previous character: Matches the string ‘book’ or ‘books’ |
| /colou?r/ | Matches the string ‘color’ or ‘colour’ |
| /beg**.**n/ | Matches any character between ‘*beg’* and ‘*n’* *Exa: begin, beg’n , begun,….* |

# Example

RE that identify the word “the” :

/ the/

Will find all strings of with ‘the as a sub string:

there, they, them, other, whether, either, another

RE that identify the word “the” at word boundary:

/ \b the \b /

# Basic Regular Expression Patterns Cont…

| 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 |
| {,m} | up to *m* occurrences of the previous char or expression |

| RE | Match |
|---|---|
| ^ | start of line |
| \$ | end of line |
| \b | word boundary |
| \B | non-word boundary |

Example: /^The dog\.$/

matches a line that contains only the phrase “The dog.”

# Example Regular Expressions

RE which will identify the strings of the form:

baa!
baaa!
baaaa!
baaaaa!
. . .

RE:

/ baaa*! /

or

/ baa+! /

# Example

- RE that identify any digit number:

  / [0-9]+ /

  or

  / [0-9][0-9]* /

- RE that identify an integer with decimal point:

  / [0-9]+ \.[0-9]+ /

- RE that identify an integer with optional decimal point:

  / [0-9]+ (\.[0-9]+)? /

- RE that identify an integer 4 decimal points:

  / [0-9]+ \.[0-9]{4} /

# Example

RE that identify the strings of the form:

Megabytes
MegaBytes
megabytes
megaBytes

RE:

/ Megabytes | MegaBytes | megabytes | megaBytes /

or

/ [Mm]ega[Bb]ytes /

# Example

Simple RE that can identify an email id of the form: sit@gmail.com

[a-z]+ @ gmail \. Com

A generic RE for all maid id domains:

[a-z]+ @ [a-z]+ \. [a-z]+

/* considering all possible character combinations.

However, this may accept invalid domain ids too*/

If valid mail ids only, create a set of all valid mail domains and use:

{gmail, yahoo, ymail, hotmail,…….}

# Homework Questions

1. Design a RE that can accept any string except the special symbols (., _, #, @, -, etc) from a text document.

2. Design a RE that can accept any string with only alphabets (upper/lower case) and a 2 digit number.

3. Design a RE that can identify any digit number with 4 decimal points followed by the strings: Megabytes, MegaBytes, megabytes, megaBytes, megabyte, MB, mb, etc

4. Design a RE that can receive a user password which must contain at least one upper case character at the beginning and combination of lowercase alphabets, at least one number and one special character,
5. If maximum 10 characters for password on Q4

# Homework Questions Cont…

6.Design a RE that can search for your mail id in text.

7. Design a generic RE that can identify any email id format from a document.
Example email id formats:
(i) sit@gmail.com
(ii) sit2020@gmail.com
(iii) sit_2020@ymail.com
(iv) sit.2000@yahoo.com
(v)amit_raj09@gmail.com
(vi) sovit_kumar@silicon.ac.in
(vii) pooja.dash09@silicon.ac.in

# Basic Regular Expression Patterns Cont…

Aliases for common sets of characters

| 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 |

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 | |

# Example

- RE that identify any Integer:

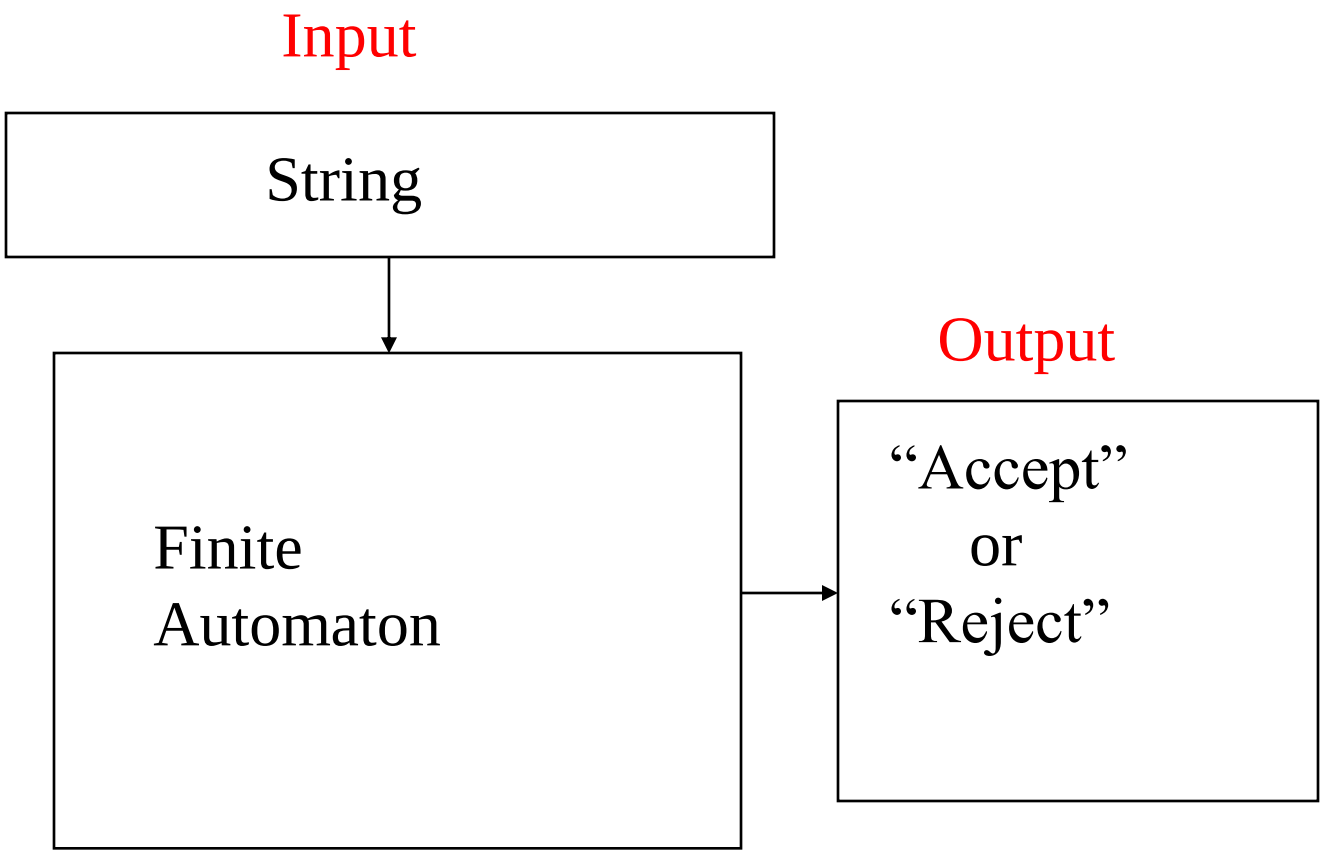
  / \d /

- RE that identify any 5 digit integer:

  / \d{5} /

# Finite-State Automata

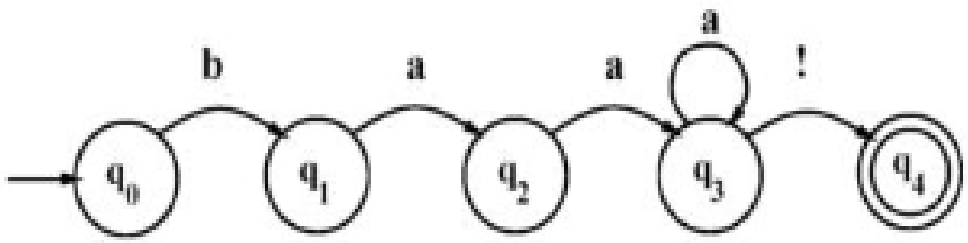
# Finite Automaton



# Example-1

baa!
baaa!
baaaa!
baaaaa!

RE: /baaa*!/ or /baa+!/

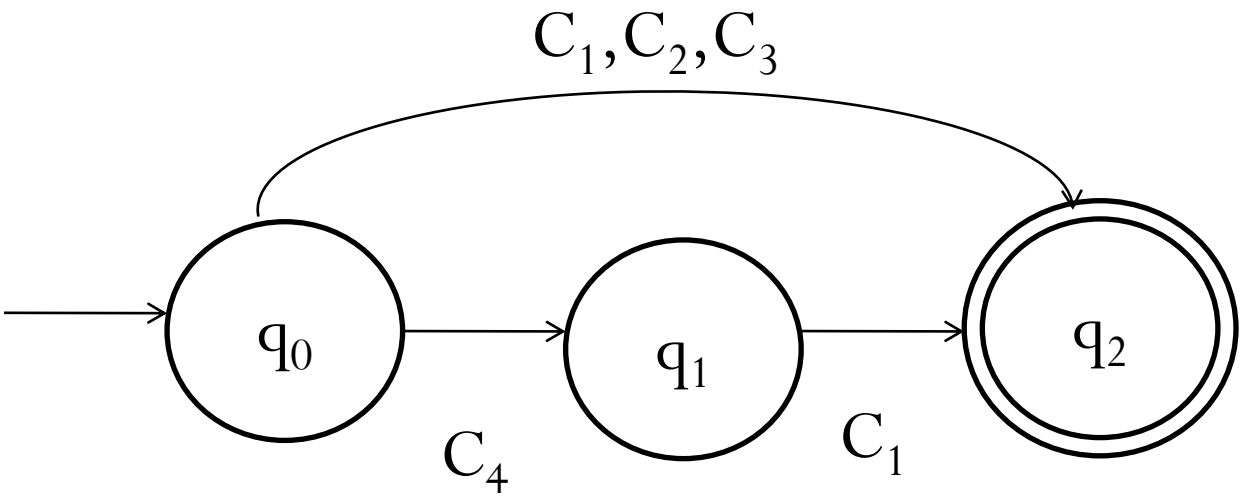



/baa+!/

# Example-2

FSA for the words of English number pronunciations 1-99

Possible pronunciation classes:
$C_1$: {one,two,……..nine}
$C_2$: {ten, twenty, ……ninety}
$C_3$:{eleven, twelve,…..,nineteen}
$C_4$: {twenty, thirty, …..ninety}



# Example-2
# alternative approach

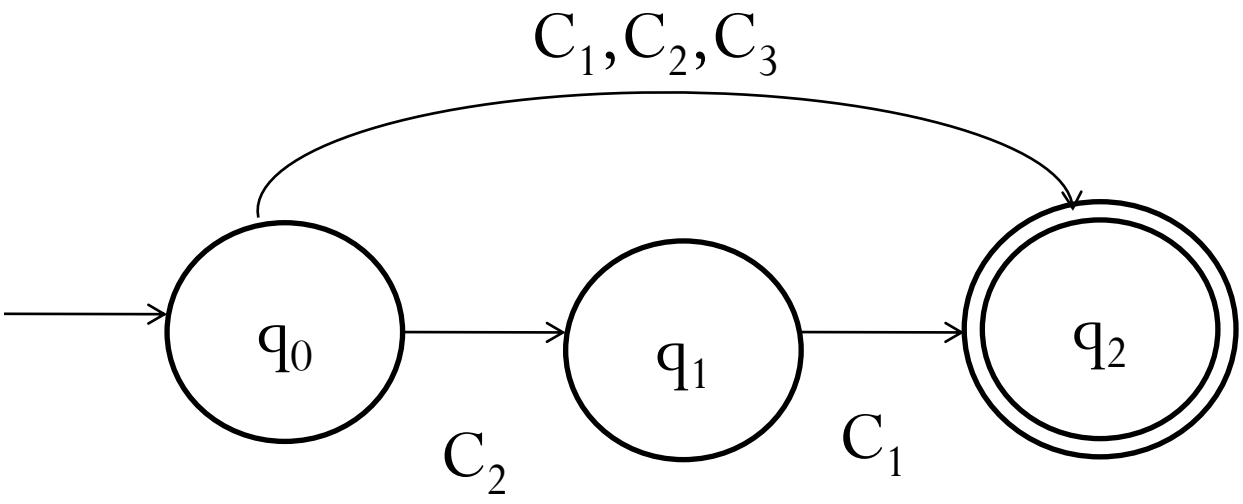
FSA for the words of English number pronunciations 1-99

Possible pronunciation classes:

$C_1$: {one, two, three,……..,nine}

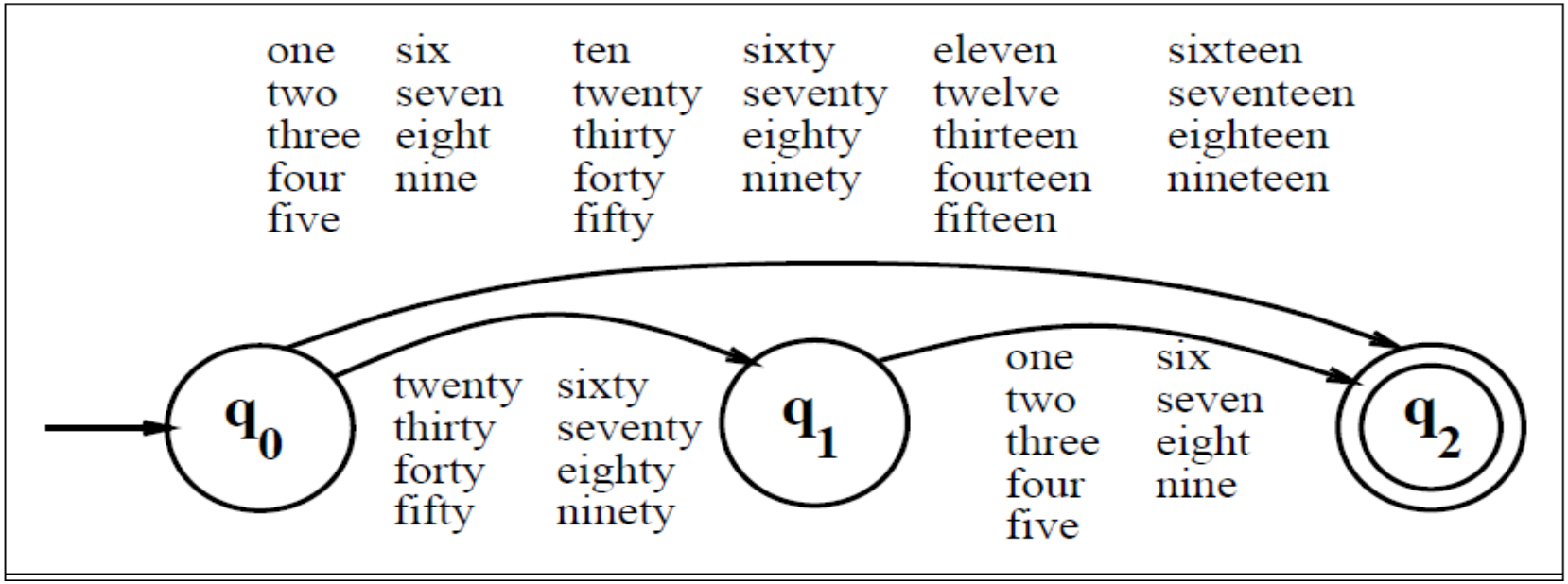
$C_2$: {twenty, thirty, forty ……,ninety}

$C_3$:{ten, eleven, twelve,…..,nineteen}



# Example-2
# alternative approach

FSA for the words of English numbers 1-99



# Example-3

FSA for the words of English numbers 1-99 with dollars and cents

Example:

20 dollars
20 dollars 35 cents
35 cents

# Example-3

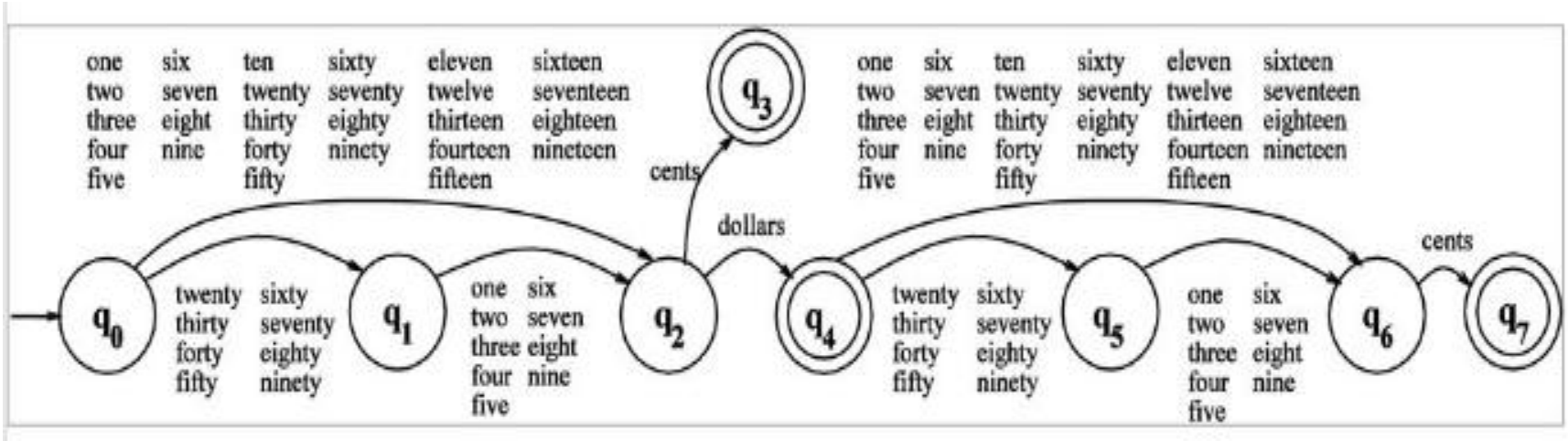
FSA for the words of English numbers 1-99 (String format) with dollars and cents

Example:

20 dollars

20 dollars 35 cents

35 cents



# Homework Questions

1. Design FSA which will accept the strings of the format: Happy Birth Day, Happy Teacher's Day, Happy Father's Day, etc.

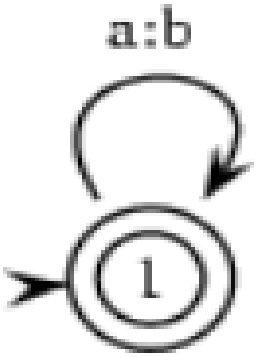
2. Design FSA which will accept the strings of the form: 1st Jan 2020, 2nd Feb 2021, 3rd Aug 2023, ……., etc in the year range 2020-2025

3. Design FSA which will accept the strings of the form: Last Monday on 21st August 2018, Next Tuesday on 14th Feb 2023, etc in the year range 2018-2025

# Finite State Transducer

- Finite State Transducers is a finite state automaton that works on two (or more) tapes.

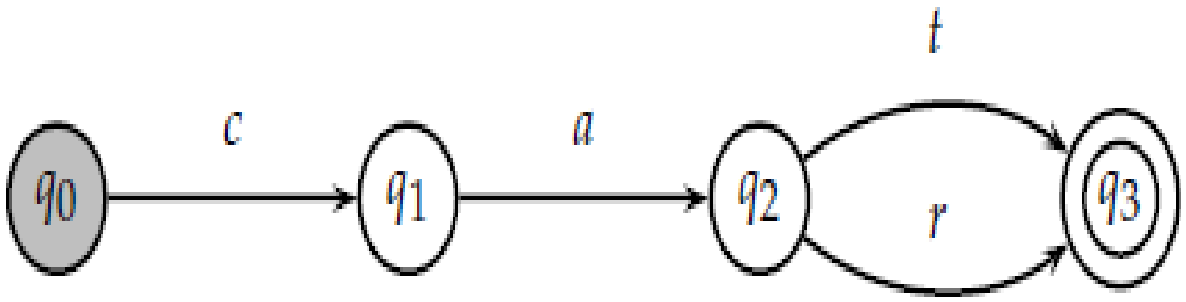
- It read from one of the tapes and write onto the other.



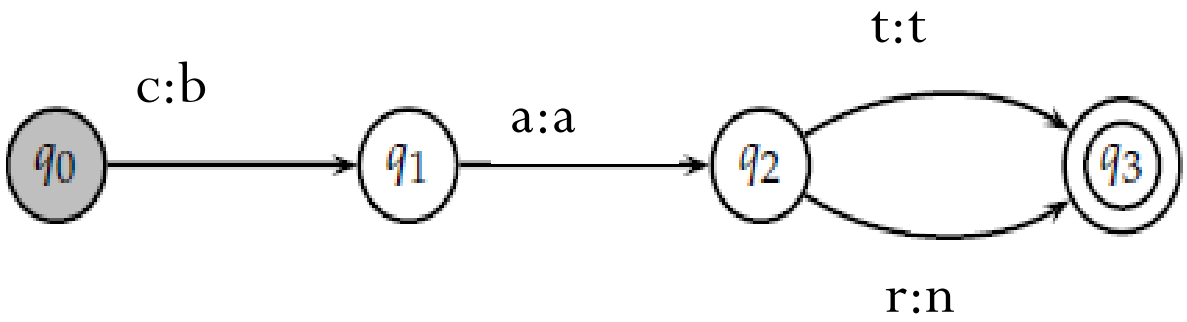

- a:b :
  - the transducer reads a from the first tape and writes b onto the second

# Example

FSA for the string :cat | car



# Example




Convert:

cat →bat

car →ban