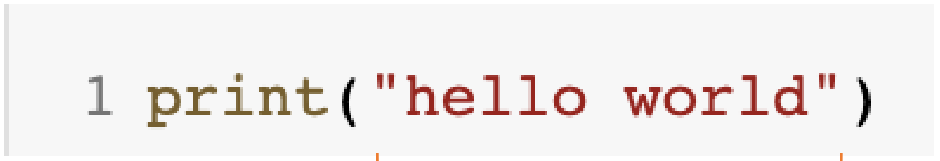
Python Libraries Assignment06

Duong Dinh Thang – TA

Outline

Review

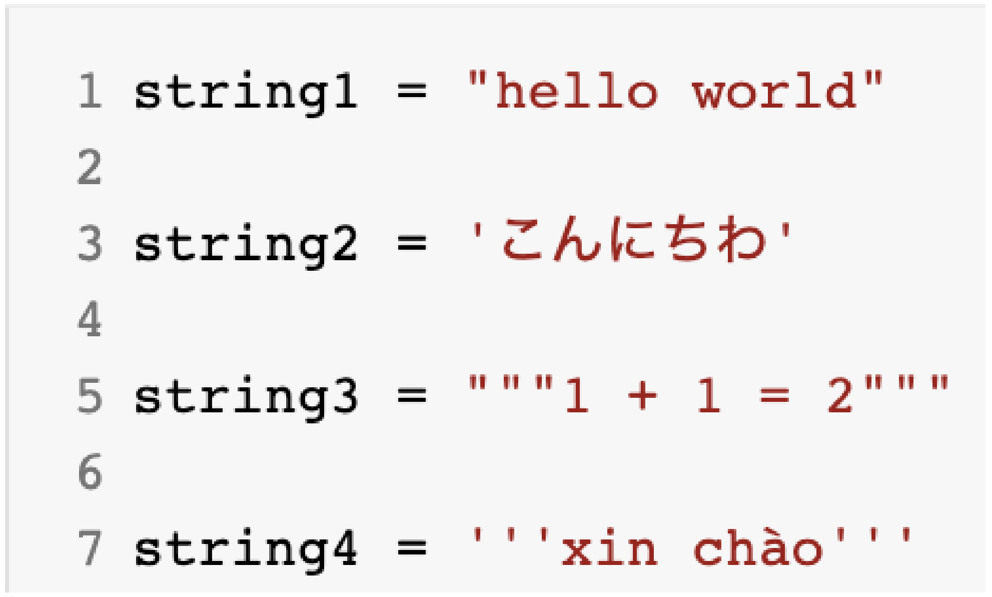
❖ String



```python
print("hello world")
```

😊 This is a *String*

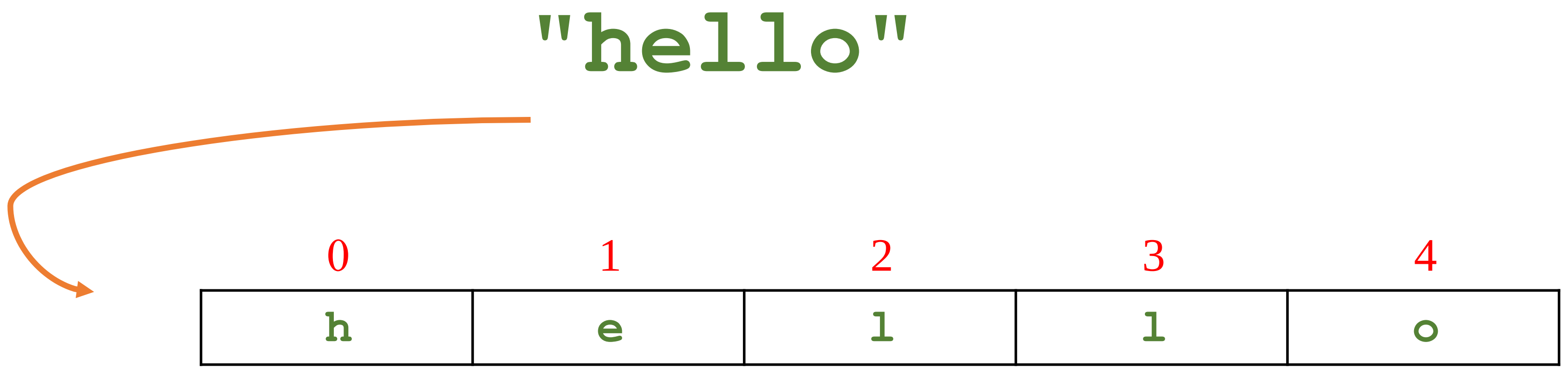
😳 *String* declaration in Python



```python
string1 = "hello world"

string2 = 'こんにちわ'

string3 = """1 + 1 = 2"""

string4 = '''xin chào'''
```

Review

❖ **String**



😬 *String* acts like a *list of characters*

☞ **indexing**

```python
string = "hello"

print(string[0])
```

```
h
```

☞ **iteration**

```python
string = "hello"
for char in string:
  print(char)
```

```
h
e
l
l
o
```

☞ **len()**

```python
string = "hello"

print(len(string))
```

```
5
```

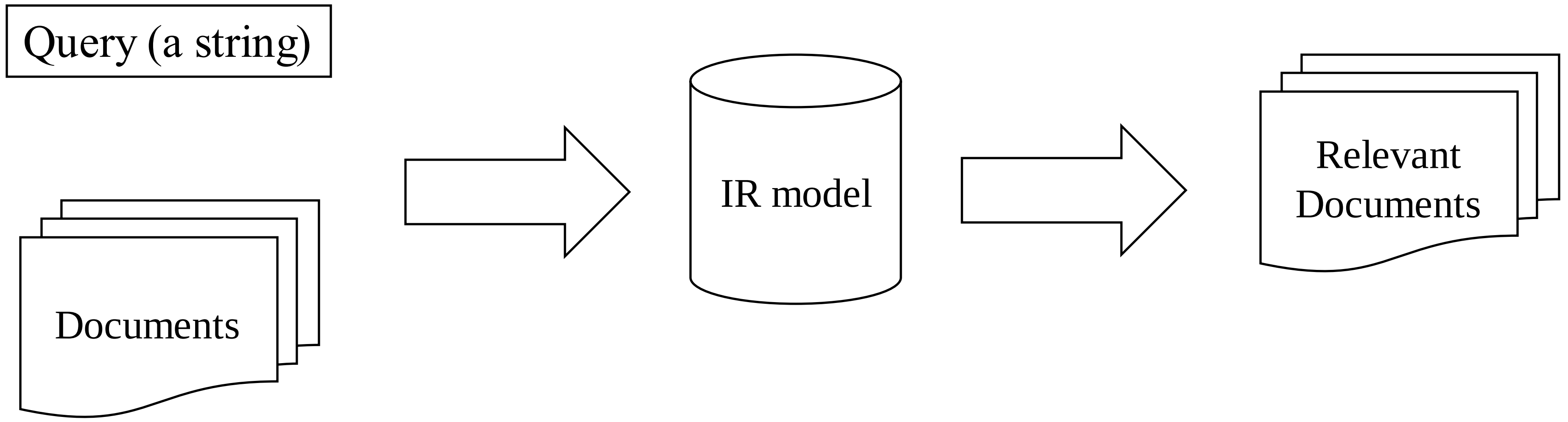
Review

❖ Essential String methods

Syntax	Definition	Example
lower()	Lowercasing string	"HELlo".lower() → "hello"
upper()	Uppercasing string	"hello".upper() → "HELLO"
replace()	Replace specified string with a new string	"hello".replace('hello', 'hi') → "hi"
join()	Concatenate strings together	" ".join(['hello', 'aio2022']) → "hello aio2022"
split()	Split a string into a list by a seperator	"Hello AIO2022".split(" ") → ['Hello', 'AIO2022']
strip()	Remove leading and trailing characters	" Hello ".strip() → "Hello"
format()	Formats string into your custom string	"Hello {name}".format(name="aio2022") → "Hello AIO2022"
startswith()	Check if a string is the initial substring of another string	"Hello AIO2022".startswith("Hello") → True

Review

❖ A task in NLP: Text Retrieval

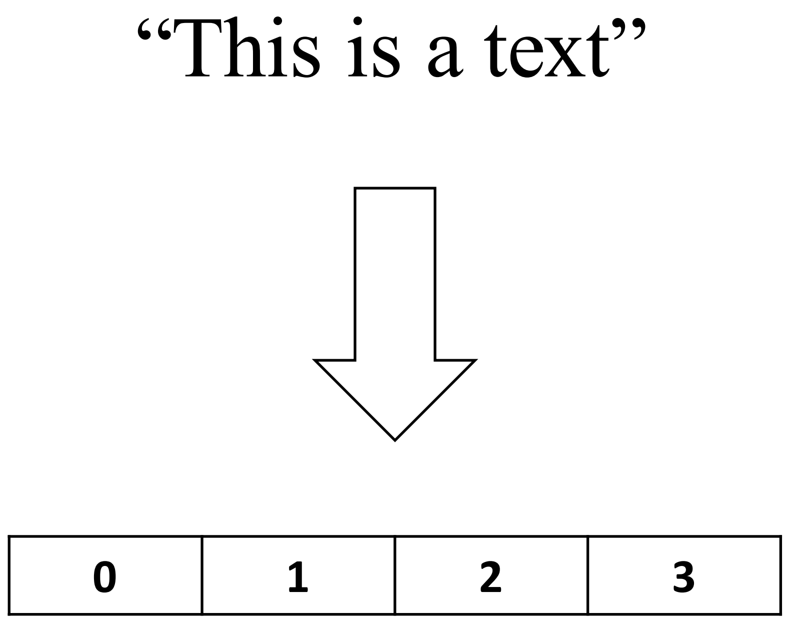


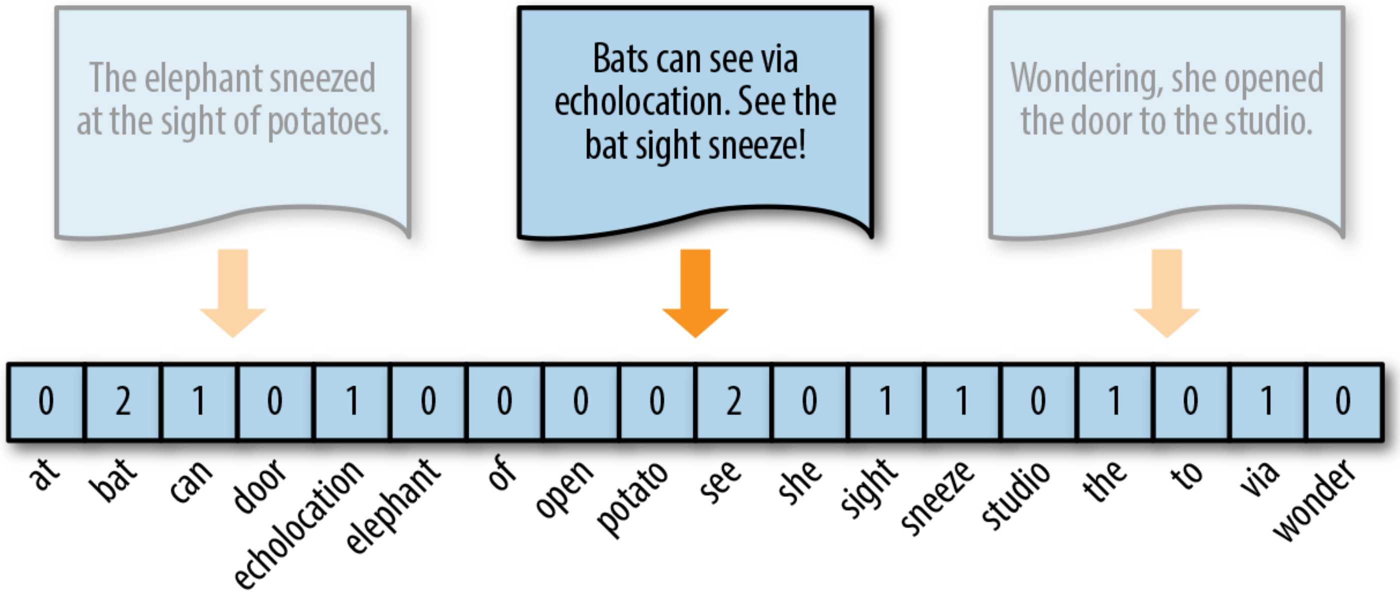
Problem:

1. How to represent string to be something that computer could calculated?
2. How to find the similarity between two string efficiently?

Review

❖ Text Vectorization



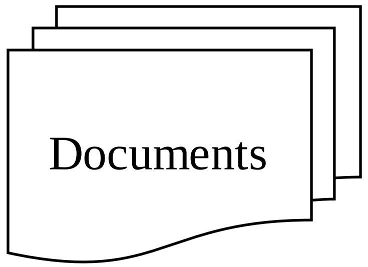


The elephant sneezed at the sight of potatoes.
Bats can see via echolocation. See the bat sight sneeze!
Wondering, she opened the door to the studio.
0 2 1 0 1 0 0 0 0 2 0 1 1 0 1 0 1 0
at
bat
can
door
echolocation
elephant
of
open
potato
see
she
sight
sneeze
studio
the
to
via
wonder

source: https://www.oreilly.com/library/view/applied-text-analysis/9781491963036/ch04.html

Review

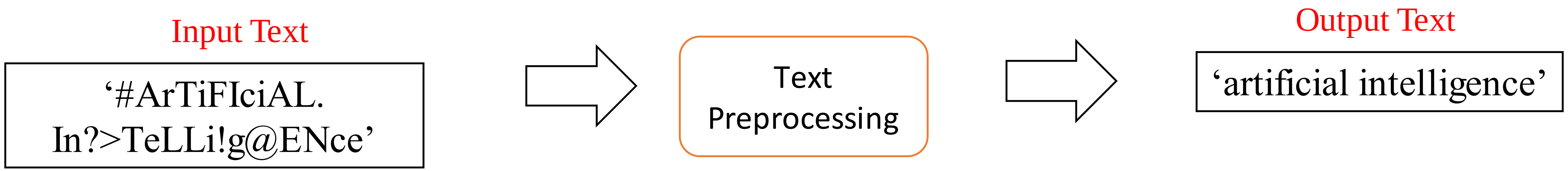
❖ Text Preprocessing



Document ID	Content
d1	Hello, we are learning information retrieval.
d2	tHIs iS a pRObLEm iN TexT ReTriEVAl
d3	#science?! <artificial intelligence> #deep learning!!!

Problem:

1. Documents contain unecessary string (information)
2. Not well-represent natural language



Problem 01

❖ Description

Abstract: build a class of *text preprocessing*, which consists of:

1. Lowercasing
2. Uppercasing
3. URL Removal
4. HTML Tags Removal
5. Punctuations Removal
6. Stopwords Removal
7. Frequent Words Removal
8. Spelling Correction
9. Stemming
10. Lemmatization

In addition to individual 10 methods for 10 techniques above , build a method that could apply all the specified techniques to an input string.

Problem 01

❖ **Lowercasing**

Lowercasing

"Hello we're AIVN"

"hello we're aivn"

Convert the given text to lowercase

Problem 01

❖ **Uppercasing**

“Hello we’re AIVN”

Uppercasing

“HELLO WE’RE AIVN”

Convert the given text to uppercase

Problem 01

❖ **URL Removal**

URL Removal

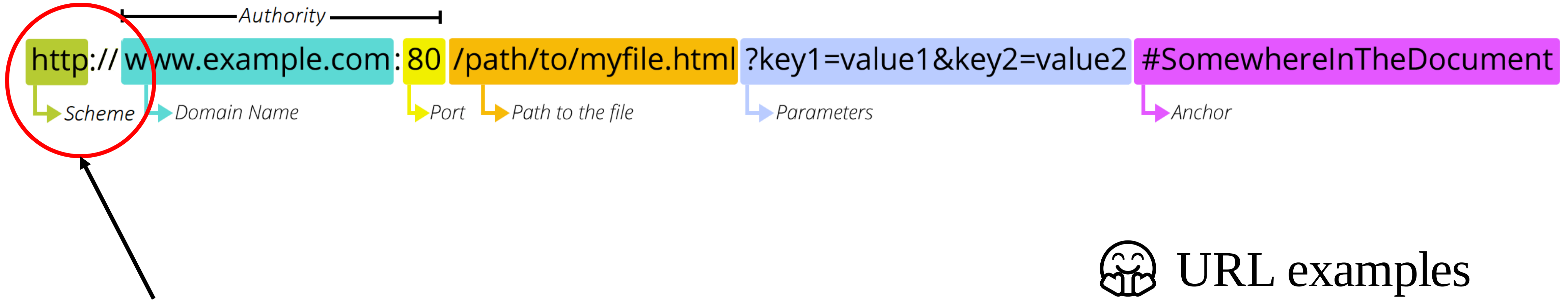
"Hello, we're AIVN.
Follow us at: https://www.facebook.com/aivietnam.edu.vn"

"Hello, we're AIVN.
Follow us at: "

Remove all URL-like substring

Problem 01

❖ URL Removal


Authority
http://www.example.com:80/path/to/myfile.html?key1=value1&key2=value2#SomewhereInTheDocument
Scheme
Domain Name
Port
Path to the file
Parameters
Anchor

Remove string starting with **http://** or **https://**

```
for word in text:
    if word.startswith('https://')
or word.startswith('http://'):
        continue
```

URL examples

https://www.google.com
https://www.facebook.com
https://www.youtube.com

source: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL

Problem 01

❖ **HTML Tags Removal**

HTML Tags Removal

“<p>Hello we are AIVN</p>”

“Hello we are AIVN”

Remove all HTML Tags in a string

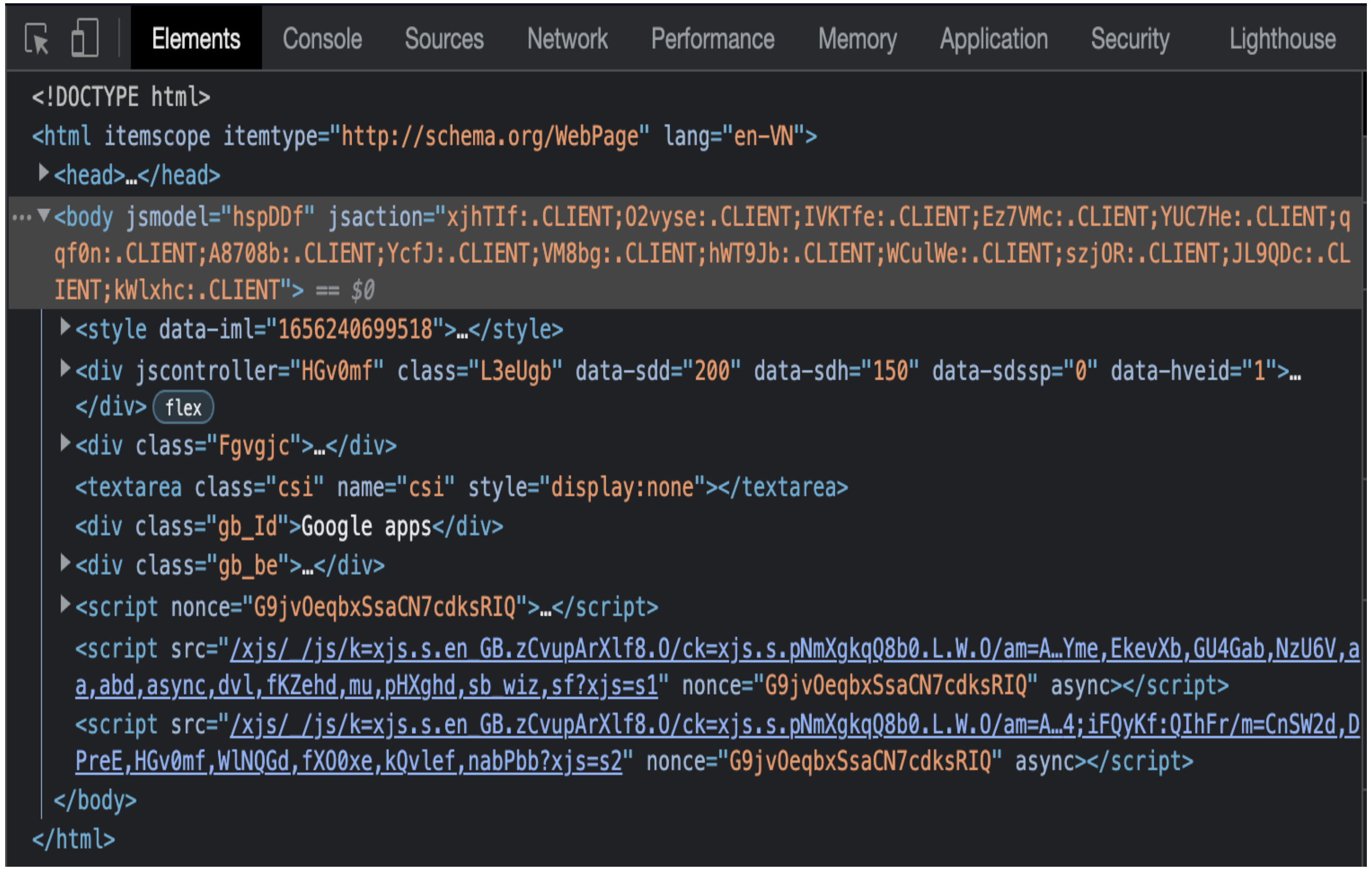
Problem 01

❖ **HTML Tags Removal**

HTML Tags string: string starts with <'text'> and end with </'text'>.

```
<html>
<body>
<h1>
Hello AIO2022
</h1>
</body>
</html>
```

HTML file example



```
<!DOCTYPE html>
<html itemscope itemtype="http://schema.org/WebPage" lang="en-VN">
 ▶<head>…</head>
…▼<body jsmodel="hspDDf" jsaction="xjhTIf:.CLIENT;O2vyse:.CLIENT;IVKTfe:.CLIENT;Ez7VMc:.CLIENT;YUC7He:.CLIENT;q
   qf0n:.CLIENT;A8708b:.CLIENT;YcfJ:.CLIENT;VM8bg:.CLIENT;hWT9Jb:.CLIENT;WCulWe:.CLIENT;szjOR:.CLIENT;JL9QDc:.CL
   IENT;kWlxhc:.CLIENT"> == $0
   ▶<style data-iml="1656240699518">…</style>
   ▶<div jscontroller="HGv0mf" class="L3eUgb" data-sdd="200" data-sdh="150" data-sdssp="0" data-hveid="1">…
    </div> flex
   ▶<div class="Fgvgjc">…</div>
    <textarea class="csi" name="csi" style="display:none"></textarea>
    <div class="gb_Id">Google apps</div>
   ▶<div class="gb_be">…</div>
   ▶<script nonce="G9jvOeqbxSsaCN7cdksRIQ">…</script>
    <script src="/xjs/_/js/k=xjs.s.en_GB.zCvupArXlf8.O/ck=xjs.s.pNmXgkqQ8b0.L.W.O/am=A…Yme,EkevXb,GU4Gab,NzU6V,a
    a,abd,async,dvl,fKZehd,mu,pHXghd,sb_wiz,sf?xjs=s1" nonce="G9jvOeqbxSsaCN7cdksRIQ" async></script>
    <script src="/xjs/_/js/k=xjs.s.en_GB.zCvupArXlf8.O/ck=xjs.s.pNmXgkqQ8b0.L.W.O/am=A…4;iFQyKf:QIhFr/m=CnSW2d,D
    PreE,HGv0mf,WlNQGd,fX00xe,kQvlef,nabPbb?xjs=s2" nonce="G9jvOeqbxSsaCN7cdksRIQ" async></script>
   </body>
</html>
```

Problem 01

❖ HTML Tags Removal

🤔 To remove HTML Tags:

🧰 Use library (BeautifulSoup):

```
from bs4 import BeautifulSoup

soup = BeautifulSoup(text)
soup.get_text()
```

✴ Use regular expression:

```
import re

pattern = re.compile('<.*?>')
re.sub(pattern, '', text)
```

Problem 01

❖ Punctuations Removal

Punctuations Removal

“Hello, welcome to AIVN.”

“Hello welcome to AIVN”

Remove all punctuations in words

Problem 01

❖ **Punctuations Removal**

Examples

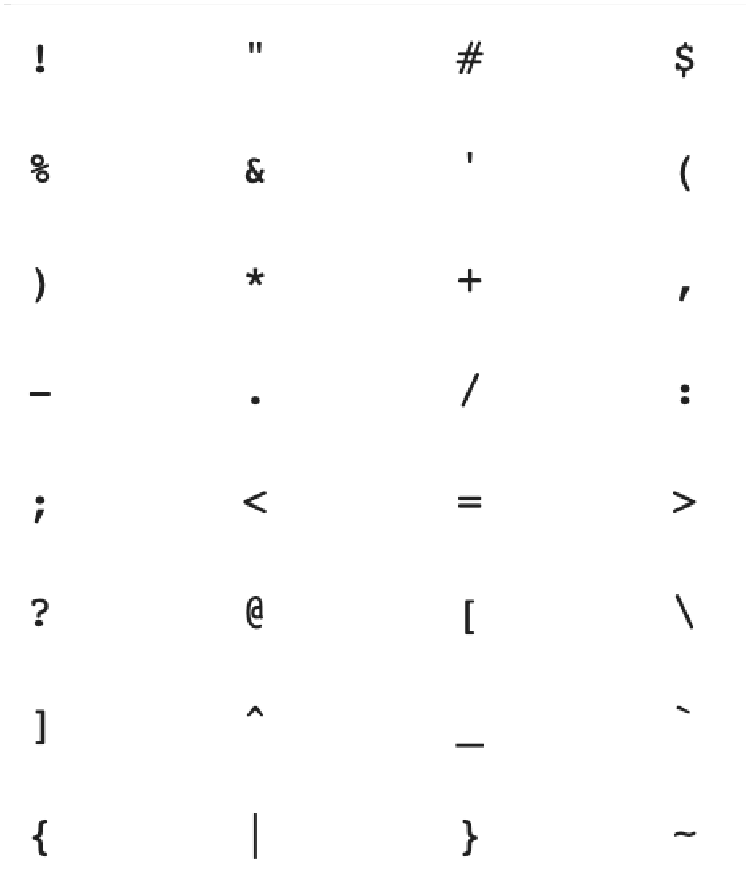
"Hello,", "#AIO2022", "we’re", "<aivn>", …

>>> "Hello", "AIO2022", "were", "aivn"

To remove:

```
import string

for word in text.split():
    for c in word:
        if c in string.punctuation:
            word.replace(c, ’’)
```

Punctuations



!	"	#	$
%	&	'	(
)	*	+	,
-	.	/	:
;	<	=	>
?	@	[	\
]	^	_	`
{	|	}	~

Problem 01

❖ **Stopwords Removal**

Stopwords Removal

"Hello we are AIVN"

"Hello AIVN"

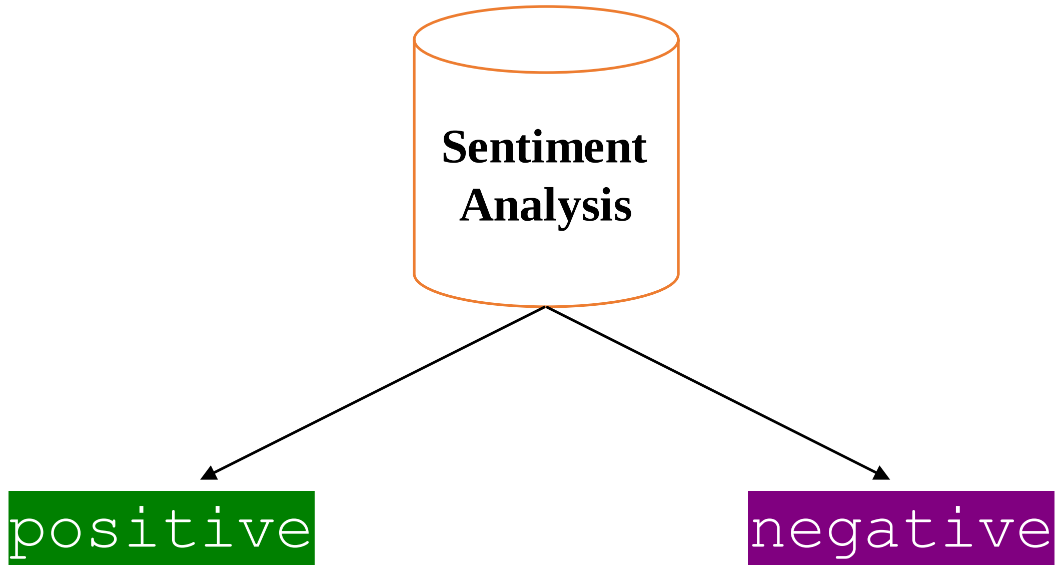
Remove all stopwords in a string

Problem 01

❖ **Stopwords Removal**

Stopwords: a set of very common words

“This is so good”



Some *Stopwords* samples

she	she's	her	hers	herself	it	it's
its	itself	they	them	their	theirs	themselves
what	which	who	whom	this	that	that'll
these	those	am	is	are	was	were
be	been	being	have	has	had	having
do	does	did	doing	a	an	the
and	but	if	or	because	as	until
while	of	at	by	for	with	about
against	between	into	through	during	before	after
above	below	to	from	up	down	in
out	on	off	over	under	again	further
then	once	here	there	when	where	why

Problem 01

❖ Stopwords Removal

🤔 To access stopwords (english):

🔗 From a raw .txt file:

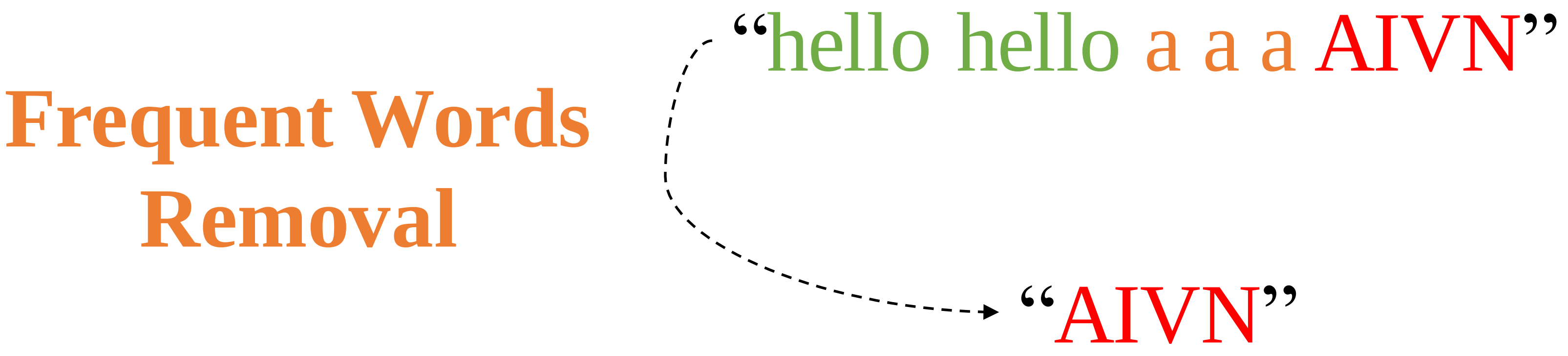
https://gist.github.com/larsyencken/1440509
https://algs4.cs.princeton.edu/35applications/stopwords.txt

📚 Use library (nltk):

```
import nltk
nltk.download("stopwords")
from nltk.corpus import stopwords

stopwords_list = list(stopwords.words("english"))
```

Problem 01

❖ **Frequent Words Removal**



Remove most frequent words in string

Problem 01

❖ Frequent Words Removal

i am learning machine learning and deep learning

Word	Count
i	1
am	1
learning	3
machine	1
and	1
deep	1

Remove 1 most frequent words in sentence:

→ i am machine and deep

Problem 01

❖ **Spelling Correction**

Spelling Correction

“Hellox, weclome to AIVN”

“Hello, welcome to AIVN”

Correct spelling of all words in string

Use library (autocorrect):

```
from autocorrect import Speller

autocorrect_spell = Speller(lang=’en’)
```

Problem 01

❖ **Stemming**

Stemming

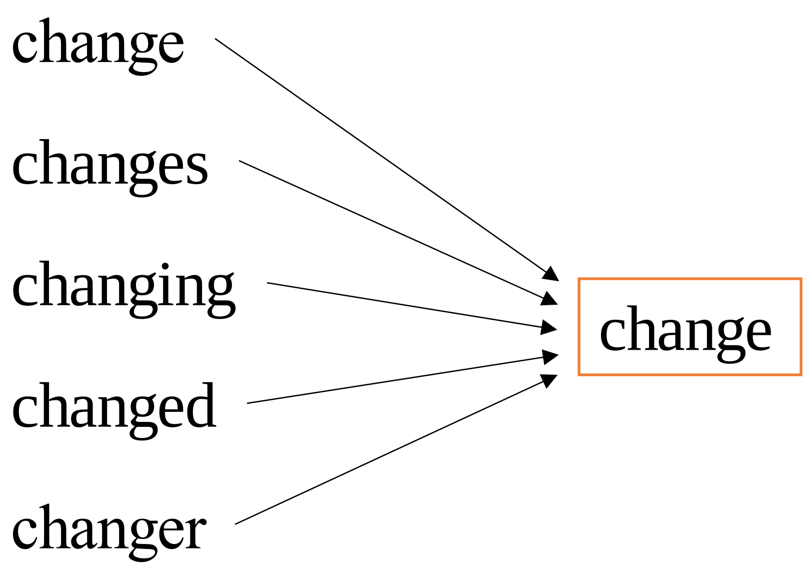
"we are learning text preprocessing"

"we are learn text preprocess"

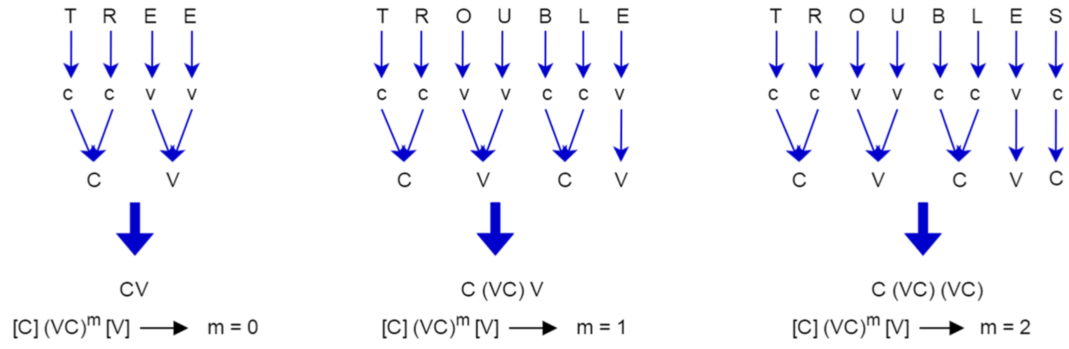
Convert word to its root form

Problem 01

❖ Stemming



stemming: convert a word to its stem form using a set of rule



Porter Stemming Algorithm

SSES	→	SS	(m>0) ATIONAL	→	ATE
IES	→	I	(m>0) TIONAL	→	TION
SS	→	SS	(m>0) ENCI	→	ENCE
S	→		(m>0) ANCI	→	ANCE

Use library

```
from nltk.stem.porter import PorterStemmer

stemmer = PorterStemmer() # stemmer.stem(word)
```

source: https://vijinimallawaarachchi.com/2017/05/09/porter-stemming-algorithm/

Problem 01

❖ **Lemmatization**

Lemmatization

"I hope you all the best"

"I hope you all the good"

Convert word to its root form

Use library (spacy):

```
import spacy

nlp = spacy.load('en_core_web_sm', disable=['parser', 'ner'])
doc = nlp(text)
# w.lemma_ for w in doc
```

Problem 02

❖ Description

Abstract: build a class of TextVectorization, which contains:

1. Tokenize a given text
2. Vectorization a tokenized text.
    1. Count Vectorizer
    2. One-hot encoding

Problem 02

❖ **Tokenization**

“This is a tokenization example”

Tokenization

[‘This’, ‘is’, ‘a’, ‘tokenization’, ‘example’]

Convert string to a list of string

Problem 02

❖ Count Vectorization

"this is a a vectorizier example"

this	that	is	are	a	vectorizer	example
1	0	1	0	2	1	1

Problem 02

❖ **One-hot Encoding**

“this is a vectorizier example”

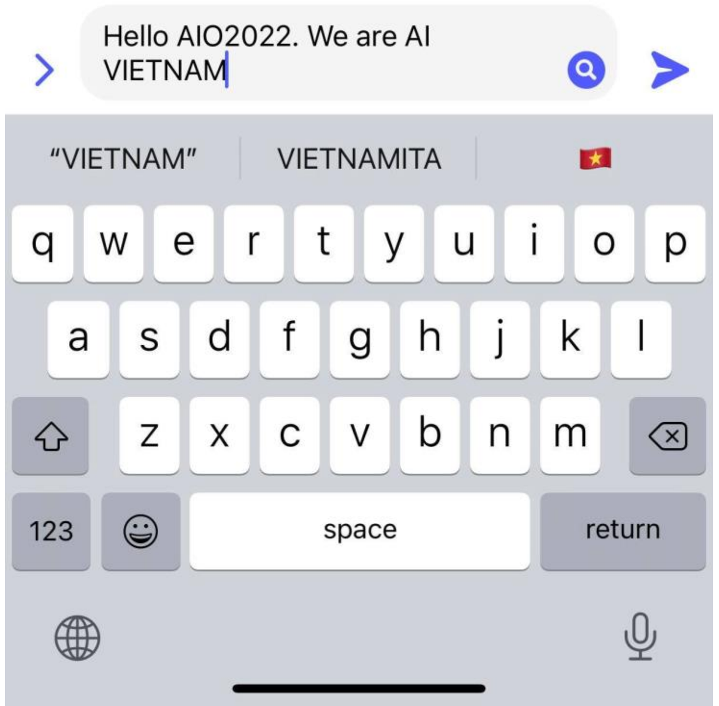
this	is	a	vectorizer	example
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Problem 03

❖ Description

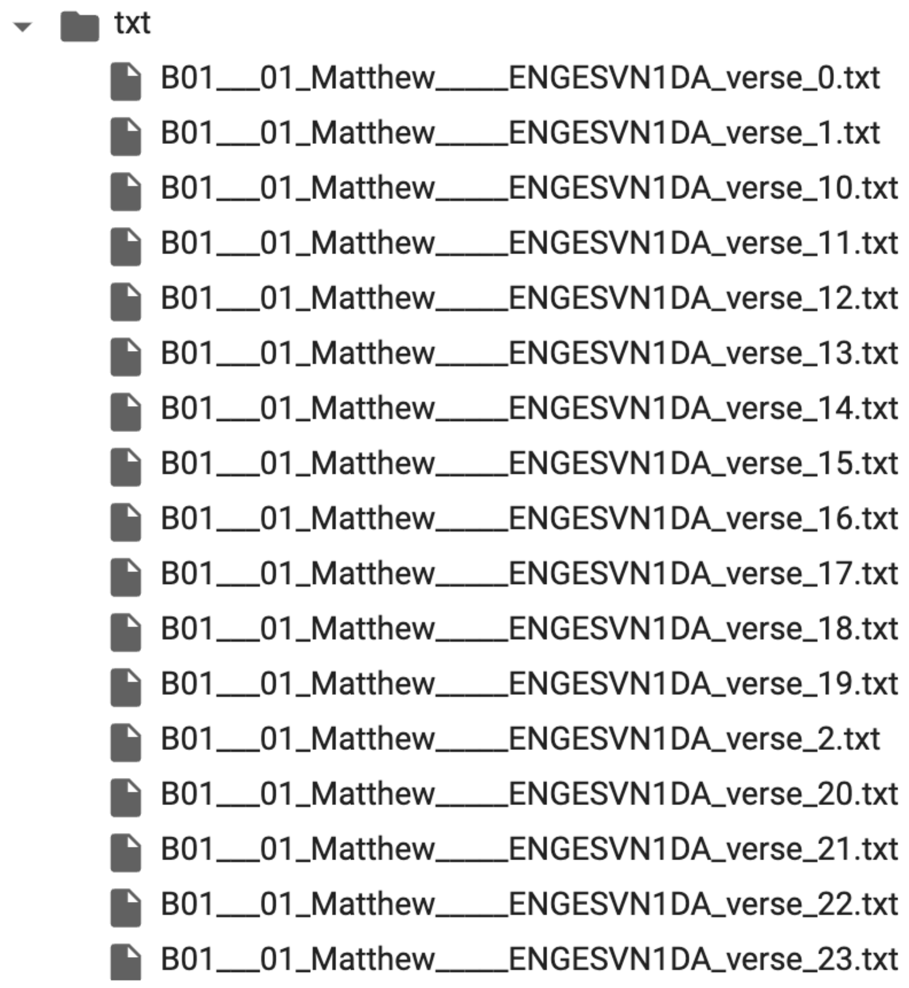
Abstract: build a program that suggest complete words given a word **using these instruction:**

1. Read all files
2. Extract unique words to create a dictionary
3. Search words that have similar starting substring of given text

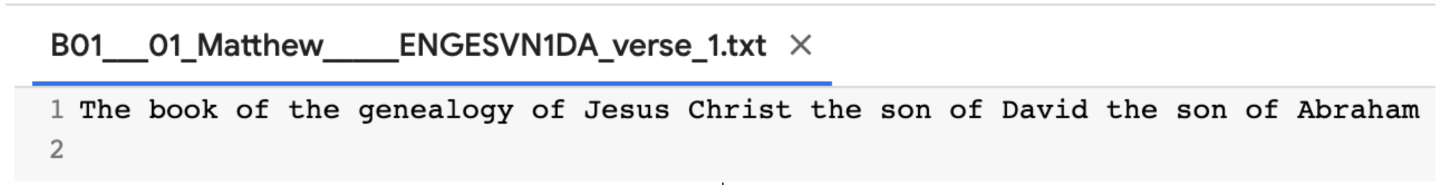

Hello AIO2022. We are AI VIETNAM

Problem 03

❖ Read files



txt
- B01___01_Matthew_____ENGESVN1DA_verse_0.txt
- B01___01_Matthew_____ENGESVN1DA_verse_1.txt
- B01___01_Matthew_____ENGESVN1DA_verse_10.txt
- B01___01_Matthew_____ENGESVN1DA_verse_11.txt
- B01___01_Matthew_____ENGESVN1DA_verse_12.txt
- B01___01_Matthew_____ENGESVN1DA_verse_13.txt
- B01___01_Matthew_____ENGESVN1DA_verse_14.txt
- B01___01_Matthew_____ENGESVN1DA_verse_15.txt
- B01___01_Matthew_____ENGESVN1DA_verse_16.txt
- B01___01_Matthew_____ENGESVN1DA_verse_17.txt
- B01___01_Matthew_____ENGESVN1DA_verse_18.txt
- B01___01_Matthew_____ENGESVN1DA_verse_19.txt
- B01___01_Matthew_____ENGESVN1DA_verse_2.txt
- B01___01_Matthew_____ENGESVN1DA_verse_20.txt
- B01___01_Matthew_____ENGESVN1DA_verse_21.txt
- B01___01_Matthew_____ENGESVN1DA_verse_22.txt
- B01___01_Matthew_____ENGESVN1DA_verse_23.txt



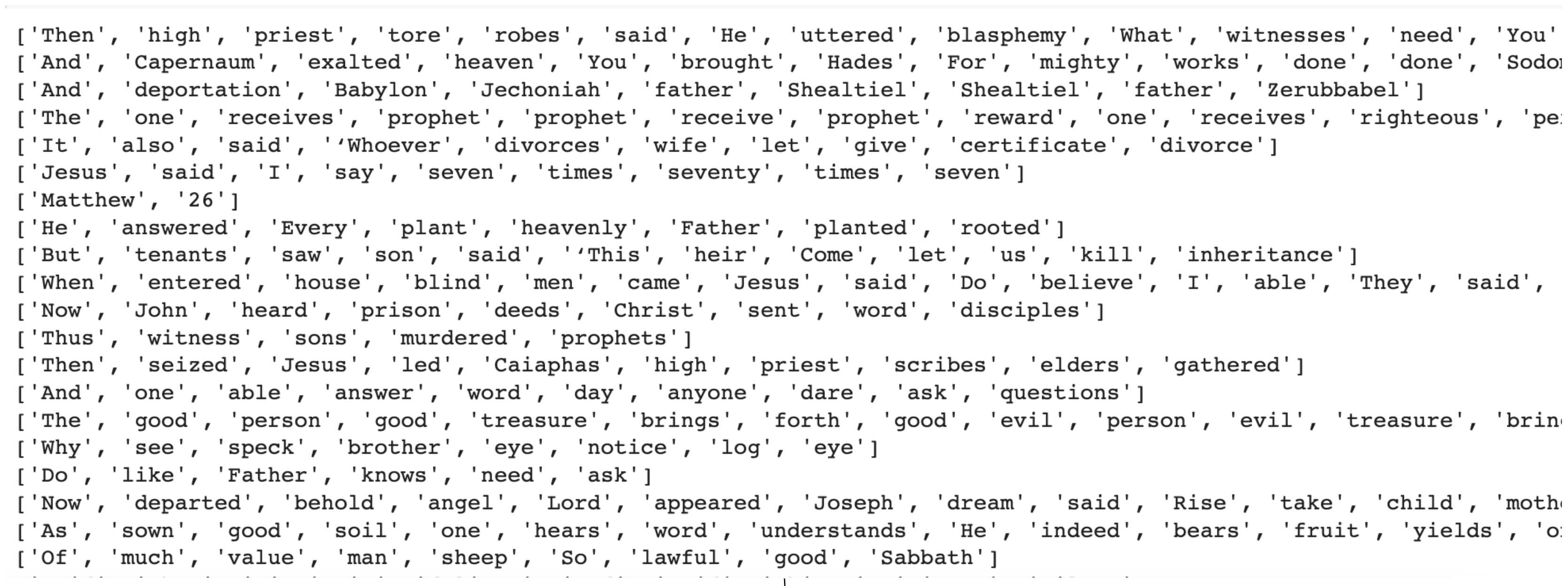
B01___01_Matthew_____ENGESVN1DA_verse_1.txt ×

```
The book of the genealogy of Jesus Christ the son of David the son of Abraham

```

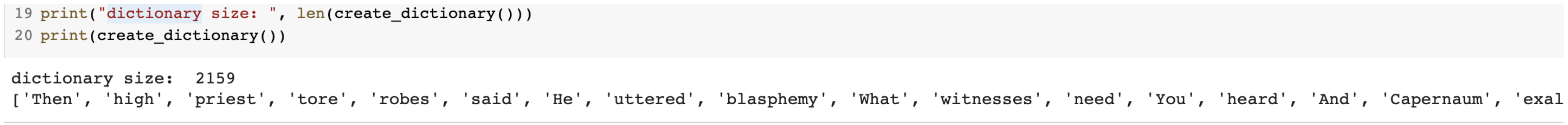
[the, book, of, the, genealogy, of, jesus…]

Problem 03

❖ Create a dictionary



```
['Then', 'high', 'priest', 'tore', 'robes', 'said', 'He', 'uttered', 'blasphemy', 'What', 'witnesses', 'need', 'You'
['And', 'Capernaum', 'exalted', 'heaven', 'You', 'brought', 'Hades', 'For', 'mighty', 'works', 'done', 'done', 'Sodo
['And', 'deportation', 'Babylon', 'Jechoniah', 'father', 'Shealtiel', 'Shealtiel', 'father', 'Zerubbabel']
['The', 'one', 'receives', 'prophet', 'prophet', 'receive', 'prophet', 'reward', 'one', 'receives', 'righteous', 'pe
['It', 'also', 'said', '‘Whoever', 'divorces', 'wife', 'let', 'give', 'certificate', 'divorce']
['Jesus', 'said', 'I', 'say', 'seven', 'times', 'seventy', 'times', 'seven']
['Matthew', '26']
['He', 'answered', 'Every', 'plant', 'heavenly', 'Father', 'planted', 'rooted']
['But', 'tenants', 'saw', 'son', 'said', '‘This', 'heir', 'Come', 'let', 'us', 'kill', 'inheritance']
['When', 'entered', 'house', 'blind', 'men', 'came', 'Jesus', 'said', 'Do', 'believe', 'I', 'able', 'They', 'said',
['Now', 'John', 'heard', 'prison', 'deeds', 'Christ', 'sent', 'word', 'disciples']
['Thus', 'witness', 'sons', 'murdered', 'prophets']
['Then', 'seized', 'Jesus', 'led', 'Caiaphas', 'high', 'priest', 'scribes', 'elders', 'gathered']
['And', 'one', 'able', 'answer', 'word', 'day', 'anyone', 'dare', 'ask', 'questions']
['The', 'good', 'person', 'good', 'treasure', 'brings', 'forth', 'good', 'evil', 'person', 'evil', 'treasure', 'brin
['Why', 'see', 'speck', 'brother', 'eye', 'notice', 'log', 'eye']
['Do', 'like', 'Father', 'knows', 'need', 'ask']
['Now', 'departed', 'behold', 'angel', 'Lord', 'appeared', 'Joseph', 'dream', 'said', 'Rise', 'take', 'child', 'moth
['As', 'sown', 'good', 'soil', 'one', 'hears', 'word', 'understands', 'He', 'indeed', 'bears', 'fruit', 'yields', 'o
['Of', 'much', 'value', 'man', 'sheep', 'So', 'lawful', 'good', 'Sabbath']
```



```python
print("dictionary size: ", len(create_dictionary()))
print(create_dictionary())
```

```
dictionary size:  2159
['Then', 'high', 'priest', 'tore', 'robes', 'said', 'He', 'uttered', 'blasphemy', 'What', 'witnesses', 'need', 'You', 'heard', 'And', 'Capernaum', 'exal
```

Problem 03

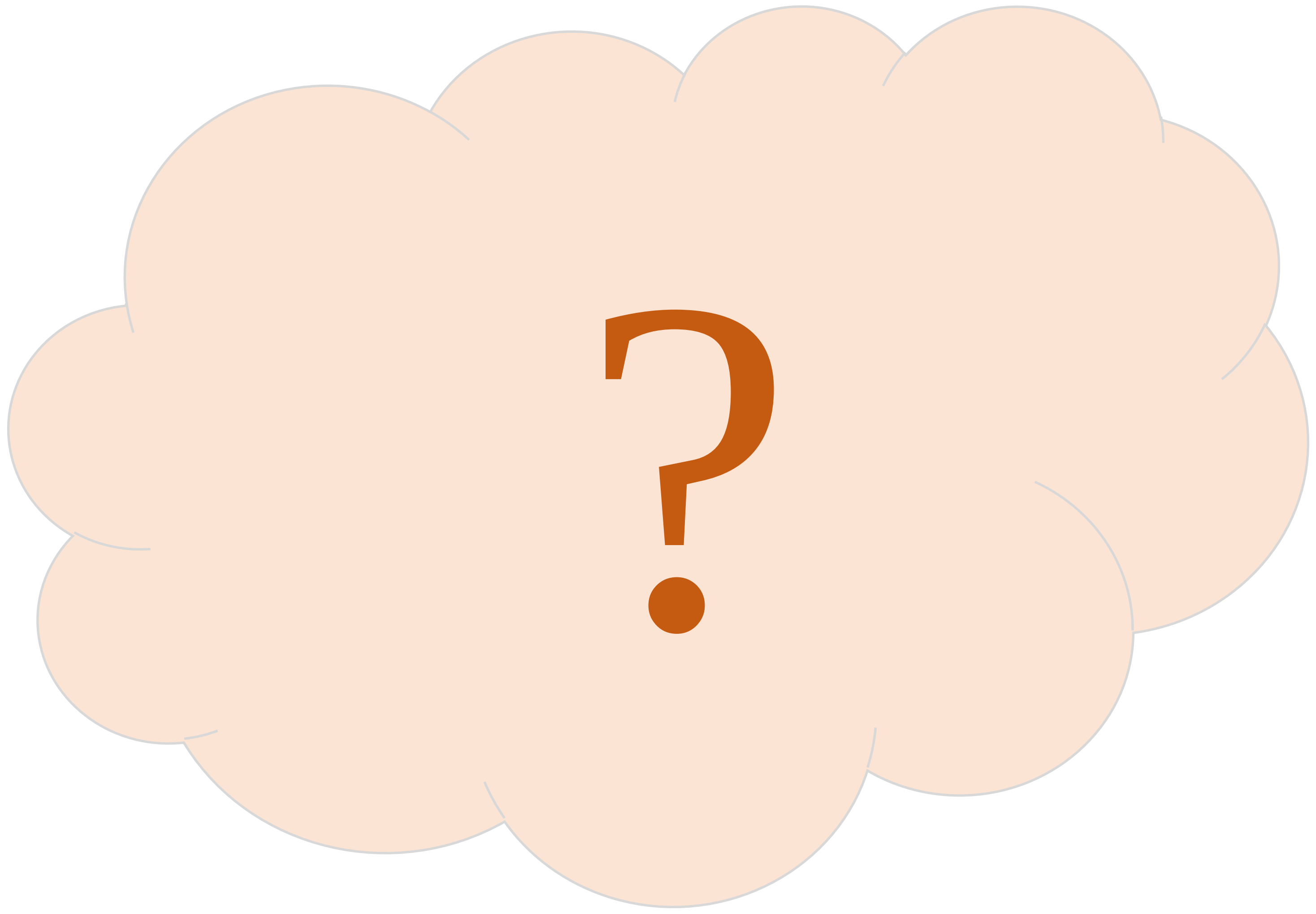
❖ **Search through dictionary to find similar string**

Words
Then
high
priest
tore
…

Search the dictionary:

```
dictionary = create_dictionary()
i = 0
print("Suggest words:")
for w in dictionary:
    if w.lower().startswith(text.lower()):
        print(f"{i + 1}. {w}")
        i += 1
```

Questions



Thank you!