MODULE 1: BUILDING LANGUAGE MODEL

The module has two sub-modules. The input to both modules is Stack Overflow data. The output of module 1 is Language model constructed using word Embedding and Inverse Document Frequency (IDF). The output of module 2 is a Knowledge Graph (KG) which is also a Language model, which is explained in later section.

Both these Language models are used to predict the results.

In the first language model, the text from the SO data is analysed and each word is vectorized. Similar words are combined and words with different meanings are kept separate. For each text the weight of each word is identified using IDF. Hence for a user query, its similar API entities can be identified.

MODULE 2: SEARCHING RELEVANT API

The functionality of this module is to generate an API rank List from the user task query using the language models. Here, the Task description query is taken as user input. In addition to the task query, the two language Models constructed in the preceding module are also used by this module.

The output is an API list with a rank for each API. The rank list implies that an API with higher rank is more relevant and is more suitable for carrying out the task specified by the user when compared to an API with relatively lower rank .

From the Word embedding model, using the keywords and key entities from the user text, the relevant Candidate APIs are selected and ranked.

Creating API ranked list for a given user query

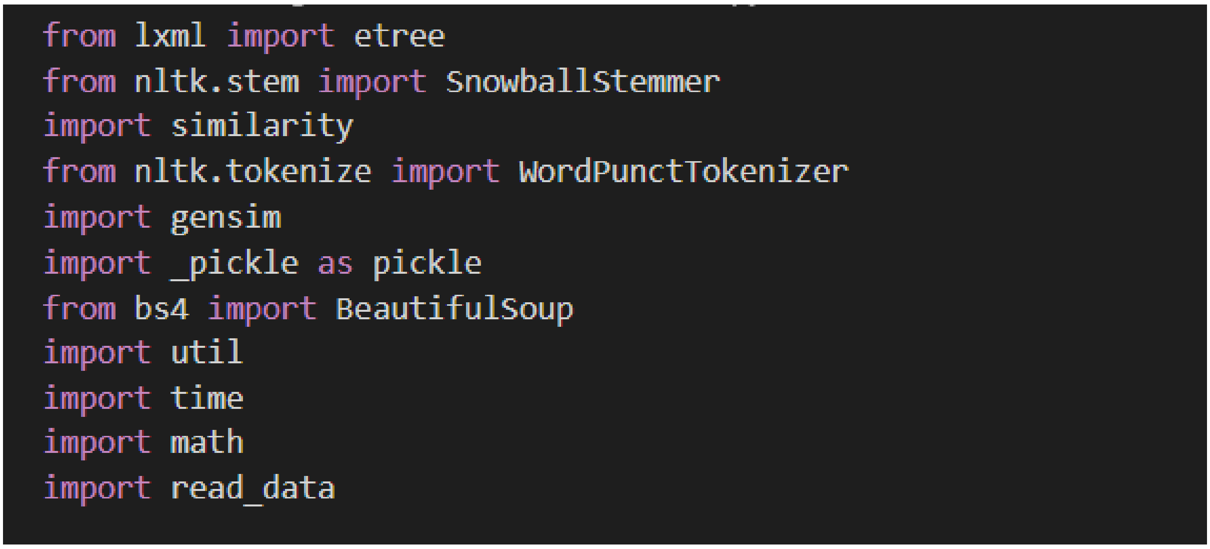
INPUT: Pre-processed data dictionary

{question : [list of apis]}

OUTPUT: Ranked API list

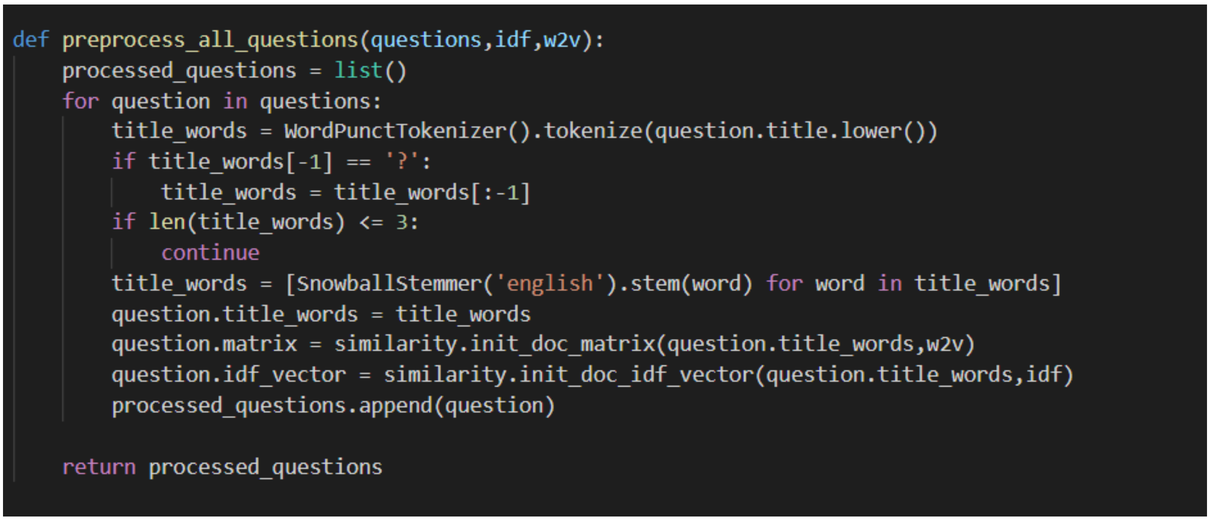
CODE:

IMPORTING LIBRARIES:



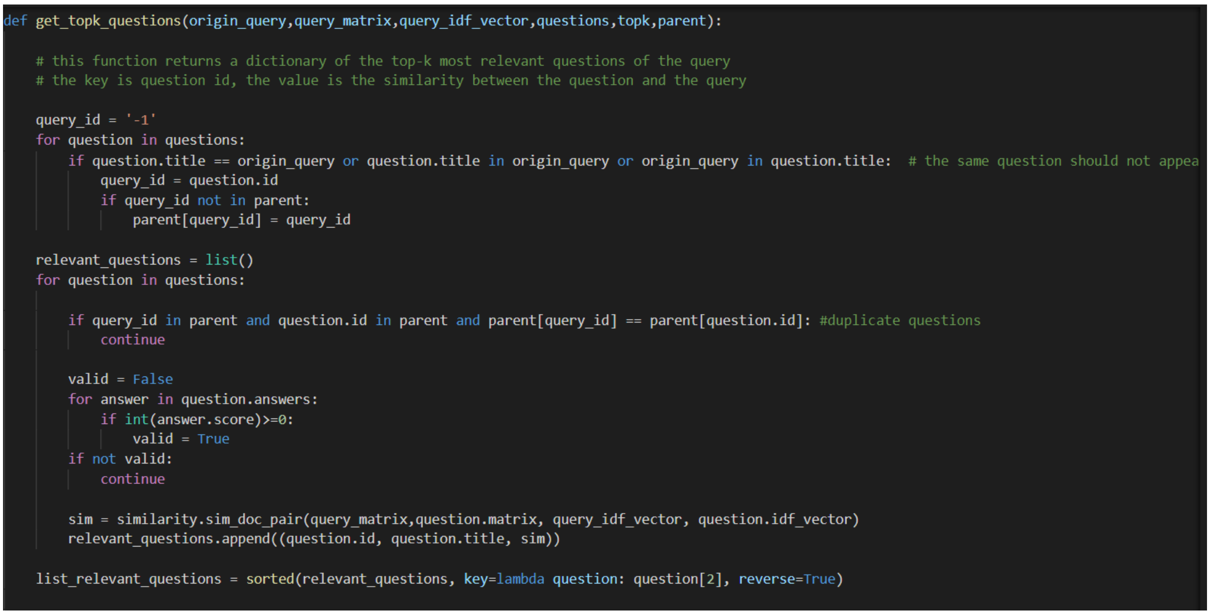
```
from lxml import etree
from nltk.stem import SnowballStemmer
import similarity
from nltk.tokenize import WordPunctTokenizer
import gensim
import _pickle as pickle
from bs4 import BeautifulSoup
import util
import time
import math
import read_data
```

GENERATING IDF VALUE , QUERY MATRIX OF ALL QUESTIONS:

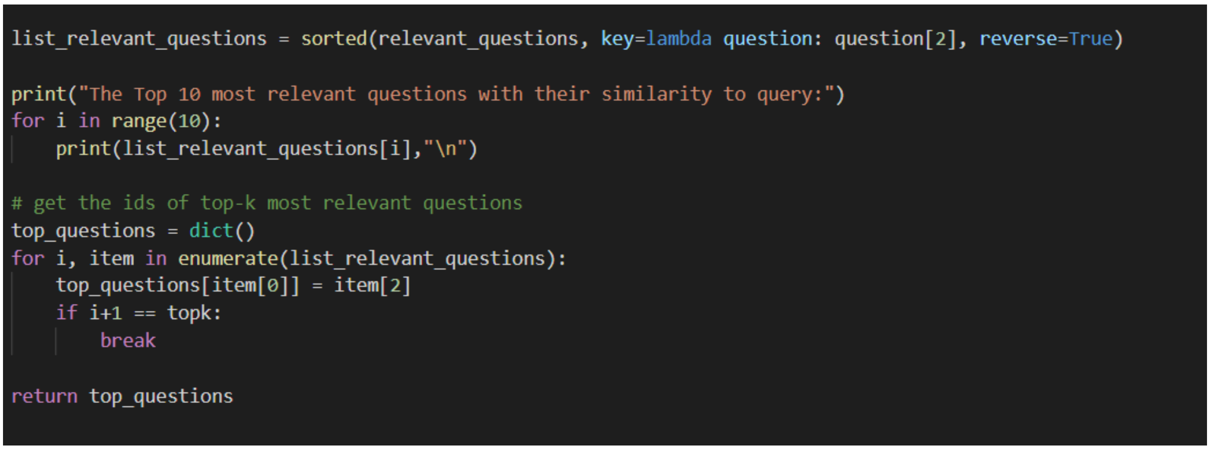


```python
def preprocess_all_questions(questions,idf,w2v):
    processed_questions = list()
    for question in questions:
        title_words = WordPunctTokenizer().tokenize(question.title.lower())
        if title_words[-1] == '?':
            title_words = title_words[:-1]
        if len(title_words) <= 3:
            continue
        title_words = [SnowballStemmer('english').stem(word) for word in title_words]
        question.title_words = title_words
        question.matrix = similarity.init_doc_matrix(question.title_words,w2v)
        question.idf_vector = similarity.init_doc_idf_vector(question.title_words,idf)
        processed_questions.append(question)

    return processed_questions
```

FILTERING TOP – K QUESTIONS:

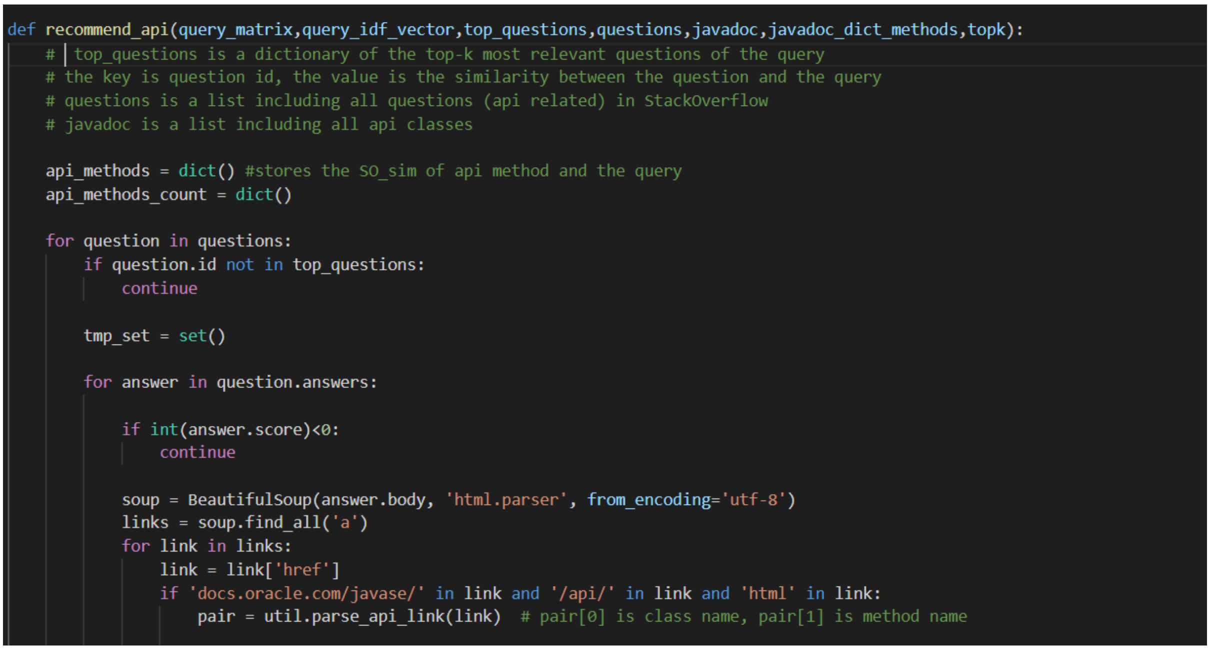


```python
def get_topk_questions(origin_query,query_matrix,query_idf_vector,questions,topk,parent):

    # this function returns a dictionary of the top-k most relevant questions of the query
    # the key is question id, the value is the similarity between the question and the query

    query_id = '-1'
    for question in questions:
        if question.title == origin_query or question.title in origin_query or origin_query in question.title:  # the same question should not appea
            query_id = question.id
            if query_id not in parent:
                parent[query_id] = query_id

    relevant_questions = list()
    for question in questions:

        if query_id in parent and question.id in parent and parent[query_id] == parent[question.id]: #duplicate questions
            continue

        valid = False
        for answer in question.answers:
            if int(answer.score)>=0:
                valid = True
        if not valid:
            continue

        sim = similarity.sim_doc_pair(query_matrix,question.matrix, query_idf_vector, question.idf_vector)
        relevant_questions.append((question.id, question.title, sim))

    list_relevant_questions = sorted(relevant_questions, key=lambda question: question[2], reverse=True)
```

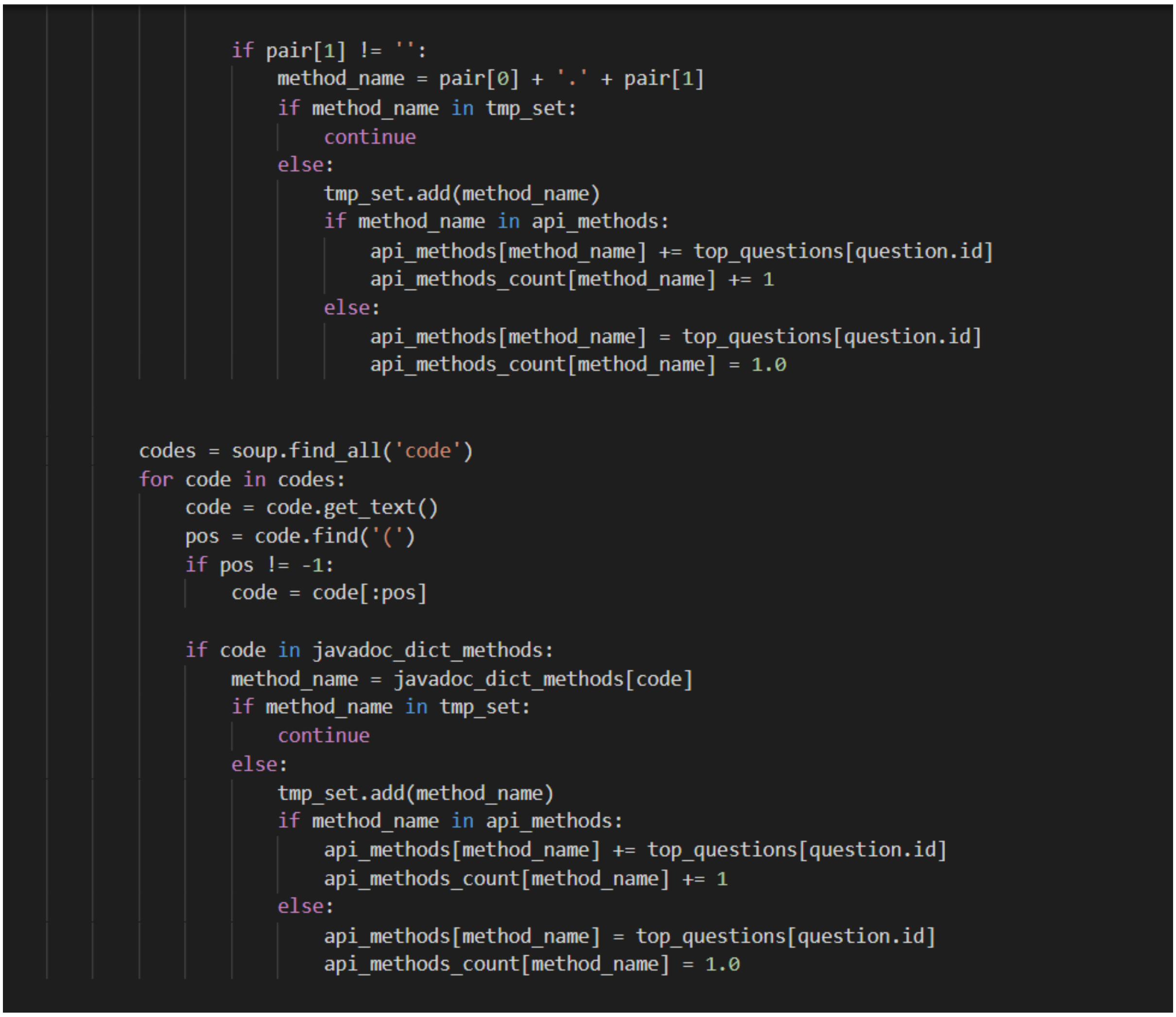


```python
list_relevant_questions = sorted(relevant_questions, key=lambda question: question[2], reverse=True)

print("The Top 10 most relevant questions with their similarity to query:")
for i in range(10):
    print(list_relevant_questions[i],"\n")

# get the ids of top-k most relevant questions
top_questions = dict()
for i, item in enumerate(list_relevant_questions):
    top_questions[item[0]] = item[2]
    if i+1 == topk:
        break

return top_questions
```

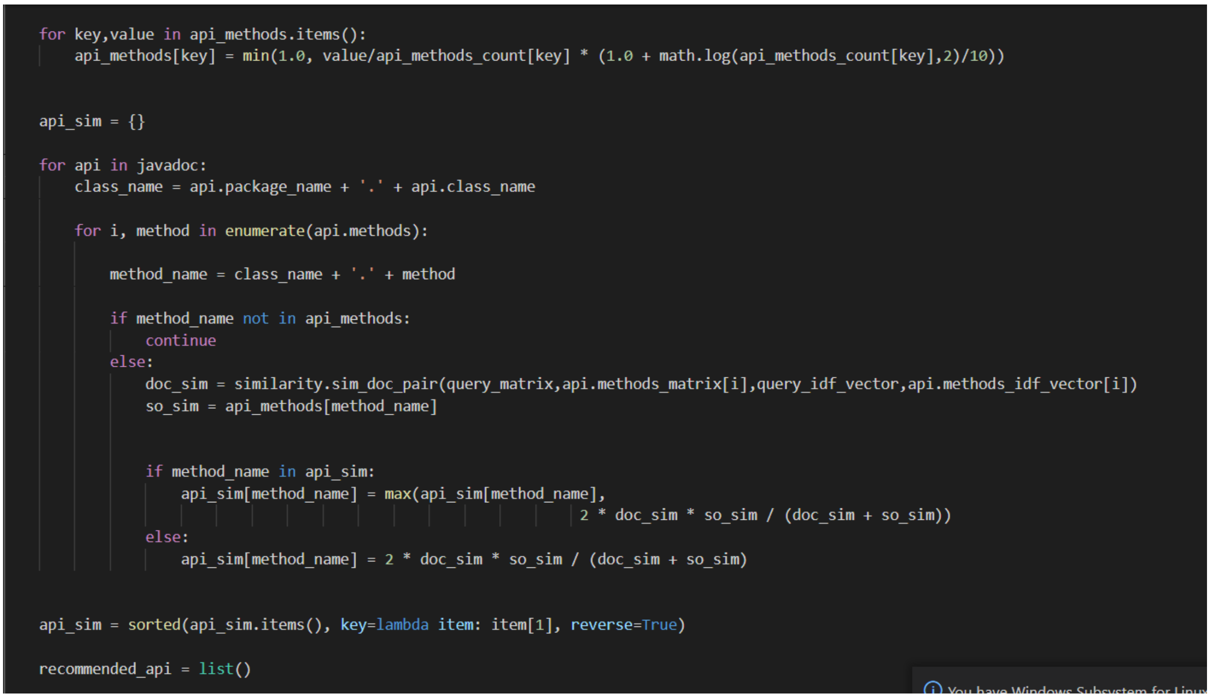
RECOMMENDATION:



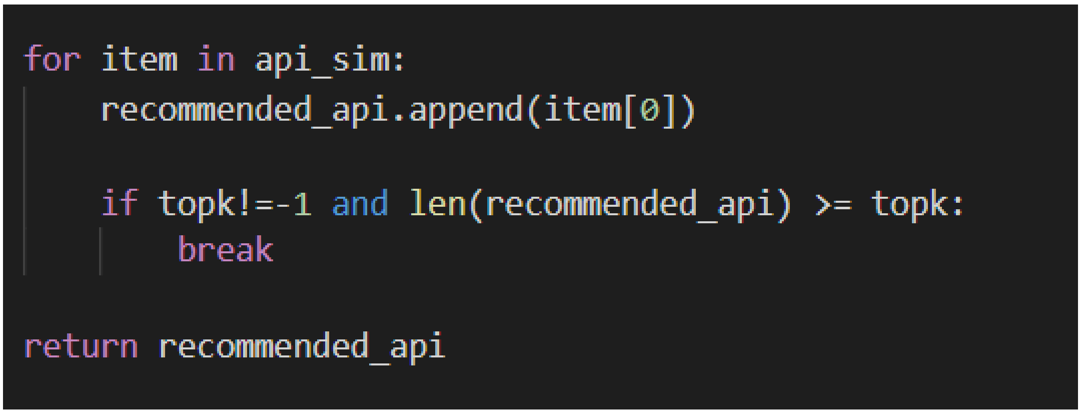
```python
def recommend_api(query_matrix,query_idf_vector,top_questions,questions,javadoc,javadoc_dict_methods,topk):
    # top_questions is a dictionary of the top-k most relevant questions of the query
    # the key is question id, the value is the similarity between the question and the query
    # questions is a list including all questions (api related) in StackOverflow
    # javadoc is a list including all api classes

    api_methods = dict() #stores the SO_sim of api method and the query
    api_methods_count = dict()

    for question in questions:
        if question.id not in top_questions:
            continue

        tmp_set = set()

        for answer in question.answers:

            if int(answer.score)<0:
                continue

            soup = BeautifulSoup(answer.body, 'html.parser', from_encoding='utf-8')
            links = soup.find_all('a')
            for link in links:
                link = link['href']
                if 'docs.oracle.com/javase/' in link and '/api/' in link and 'html' in link:
                    pair = util.parse_api_link(link)  # pair[0] is class name, pair[1] is method name
```



```python
                    if pair[1] != '':
                        method_name = pair[0] + '.' + pair[1]
                        if method_name in tmp_set:
                            continue
                        else:
                            tmp_set.add(method_name)
                            if method_name in api_methods:
                                api_methods[method_name] += top_questions[question.id]
                                api_methods_count[method_name] += 1
                            else:
                                api_methods[method_name] = top_questions[question.id]
                                api_methods_count[method_name] = 1.0


            codes = soup.find_all('code')
            for code in codes:
                code = code.get_text()
                pos = code.find('(')
                if pos != -1:
                    code = code[:pos]

                if code in javadoc_dict_methods:
                    method_name = javadoc_dict_methods[code]
                    if method_name in tmp_set:
                        continue
                    else:
                        tmp_set.add(method_name)
                        if method_name in api_methods:
                            api_methods[method_name] += top_questions[question.id]
                            api_methods_count[method_name] += 1
                        else:
                            api_methods[method_name] = top_questions[question.id]
                            api_methods_count[method_name] = 1.0
```



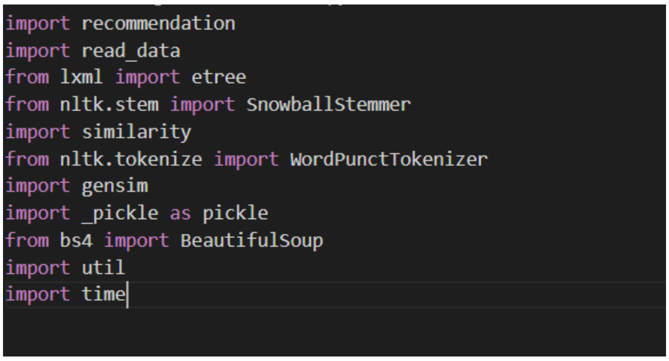
```python
for key,value in api_methods.items():
    api_methods[key] = min(1.0, value/api_methods_count[key] * (1.0 + math.log(api_methods_count[key],2)/10))


api_sim = {}

for api in javadoc:
    class_name = api.package_name + '.' + api.class_name

    for i, method in enumerate(api.methods):

        method_name = class_name + '.' + method

        if method_name not in api_methods:
            continue
        else:
            doc_sim = similarity.sim_doc_pair(query_matrix,api.methods_matrix[i],query_idf_vector,api.methods_idf_vector[i])
            so_sim = api_methods[method_name]


            if method_name in api_sim:
                api_sim[method_name] = max(api_sim[method_name],
                                           2 * doc_sim * so_sim / (doc_sim + so_sim))
            else:
                api_sim[method_name] = 2 * doc_sim * so_sim / (doc_sim + so_sim)


api_sim = sorted(api_sim.items(), key=lambda item: item[1], reverse=True)

recommended_api = list()
```



```python
for item in api_sim:
    recommended_api.append(item[0])

    if topk!=-1 and len(recommended_api) >= topk:
        break

return recommended_api
```

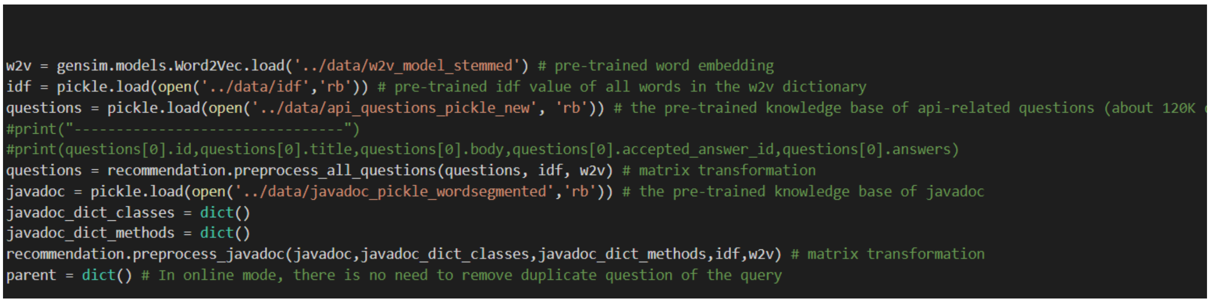
DRIVER MODULE:

IMPORTS:



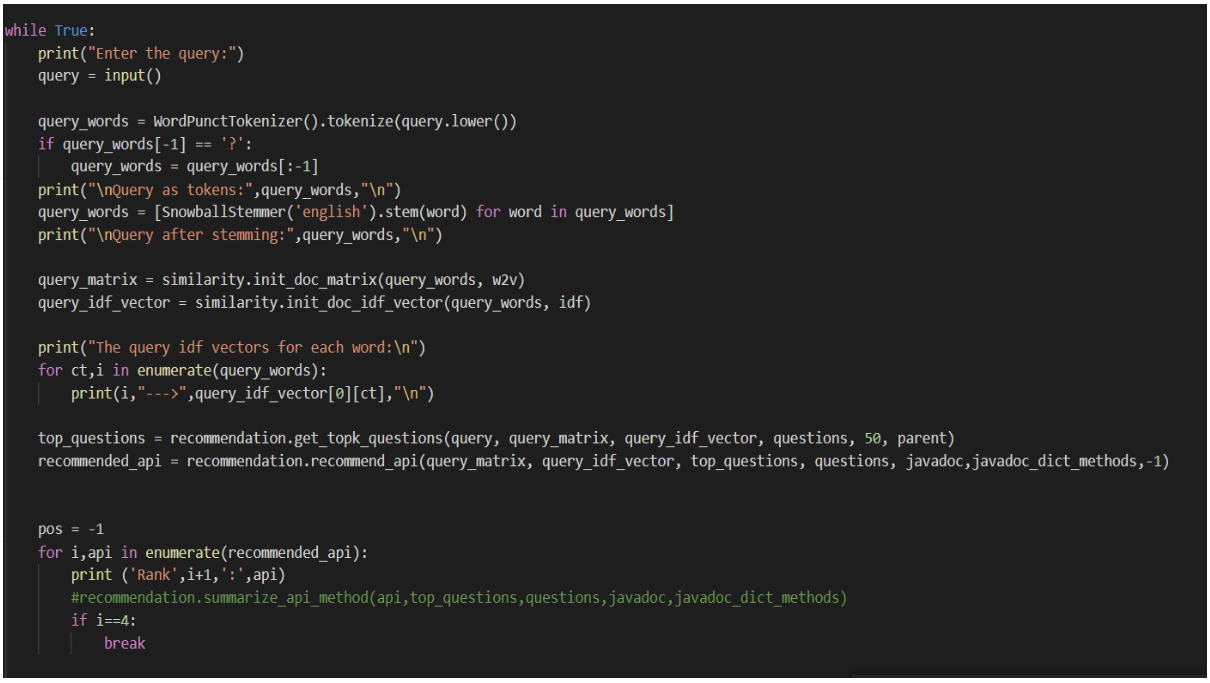
```python
import recommendation
import read_data
from lxml import etree
from nltk.stem import SnowballStemmer
import similarity
from nltk.tokenize import WordPunctTokenizer
import gensim
import _pickle as pickle
from bs4 import BeautifulSoup
import util
import time
```

LOADING DATA MODELS:



```
w2v = gensim.models.Word2Vec.load('../data/w2v_model_stemmed') # pre-trained word embedding
idf = pickle.load(open('../data/idf','rb')) # pre-trained idf value of all words in the w2v dictionary
questions = pickle.load(open('../data/api_questions_pickle_new', 'rb')) # the pre-trained knowledge base of api-related questions (about 120K
#print("-------------------------------")
#print(questions[0].id,questions[0].title,questions[0].body,questions[0].accepted_answer_id,questions[0].answers)
questions = recommendation.preprocess_all_questions(questions, idf, w2v) # matrix transformation
javadoc = pickle.load(open('../data/javadoc_pickle_wordsegmented','rb')) # the pre-trained knowledge base of javadoc
javadoc_dict_classes = dict()
javadoc_dict_methods = dict()
recommendation.preprocess_javadoc(javadoc,javadoc_dict_classes,javadoc_dict_methods,idf,w2v) # matrix transformation
parent = dict() # In online mode, there is no need to remove duplicate question of the query
```

QUERY PROCESSING:



```
while True:
    print("Enter the query:")
    query = input()

    query_words = WordPunctTokenizer().tokenize(query.lower())
    if query_words[-1] == '?':
        query_words = query_words[:-1]
    print("\nQuery as tokens:",query_words,"\n")
    query_words = [SnowballStemmer('english').stem(word) for word in query_words]
    print("\nQuery after stemming:",query_words,"\n")

    query_matrix = similarity.init_doc_matrix(query_words, w2v)
    query_idf_vector = similarity.init_doc_idf_vector(query_words, idf)

    print("The query idf vectors for each word:\n")
    for ct,i in enumerate(query_words):
        print(i,"--->",query_idf_vector[0][ct],"\n")

    top_questions = recommendation.get_topk_questions(query, query_matrix, query_idf_vector, questions, 50, parent)
    recommended_api = recommendation.recommend_api(query_matrix, query_idf_vector, top_questions, questions, javadoc,javadoc_dict_methods,-1)


    pos = -1
    for i,api in enumerate(recommended_api):
        print ('Rank',i+1,':',api)
        #recommendation.summarize_api_method(api,top_questions,questions,javadoc,javadoc_dict_methods)
        if i==4:
            break
```

Test Input:

QUERY
How to get input from console in java?
How to convert string to integer?
How to convert integer to string?
How to write in a file in java?
How to create a thread in java?
How to kill a thread ?
How to store large integer in java?
How to create a stack?
How to catch a exception?
How to collect garbage in java?
How to get integer input in java?
how to print stacktrace in java?

OUTPUT:

QUERY	TOP 5 RANKED APIs
How to get input from console in java?	Rank 1 : java.util.Scanner.next Rank 2 : java.util.Scanner.nextLine Rank 3 : java.lang.System.console Rank 4 : java.io.BufferedReader.readLine Rank 5 : java.io.FilterOutputStream.write
How to convert string to integer?	Rank 1 : java.lang.String.toCharArray Rank 2 : java.lang.Integer.toString Rank 3 : java.lang.Integer.valueOf Rank 4 : java.lang.Double.toString Rank 5 : java.lang.Integer.intValue
How to convert integer to string?	Rank 1 : java.lang.String.toCharArray Rank 2 : java.lang.Integer.toString Rank 3 : java.lang.Integer.valueOf Rank 4 : java.lang.Double.toString Rank 5 : java.lang.Integer.intValue
How to write in a file in java?	Rank 1 : java.nio.file.Files.newBufferedWriter Rank 2 : java.nio.file.Files.write Rank 3 : java.nio.file.Files.isWritable Rank 4 : javax.imageio.ImageIO.write Rank 5 : java.nio.file.Files.createDirectories

How to create a thread in java?	Rank 1 : java.util.concurrent.Executors.newCachedThreadPool Rank 2 : java.lang.Thread.join Rank 3 : java.util.concurrent.Executors.newSingleThreadExecutor Rank 4 : java.lang.Thread.currentThread Rank 5 : java.lang.Thread.start
How to kill a thread ?	Rank 1 : java.lang.Process.destroy Rank 2 : java.lang.Thread.join Rank 3 : java.lang.Thread.stop Rank 4 : java.util.concurrent.ExecutorService.shutdownNow Rank 5 : java.util.concurrent.ThreadPoolExecutor.shutdownNow
How to store large integer in java?	Rank 1 : java.lang.Integer.valueOf Rank 2 : java.lang.Long.valueOf Rank 3 : java.lang.Double.parseDouble Rank 4 : java.lang.Integer.parseInt Rank 5 : java.lang.Math.round
How to create a stack?	Rank 1 : java.lang.Thread.dumpStack Rank 2 : java.lang.Throwable.getStackTrace Rank 3 : java.lang.Throwable.fillInStackTrace Rank 4 : java.lang.Thread.getStackTrace Rank 5 : java.lang.Thread.getAllStackTraces

How to catch a exception?	Rank 1 : java.lang.Throwable.getCause Rank 2 : java.util.Scanner.nextDouble Rank 3 : java.io.File.isFile Rank 4 : java.io.File.isDirectory Rank 5 : java.io.FileInputStream.read
How to collect garbage in java?	Rank 1 : java.lang.System.gc Rank 2 : java.lang.Runtime.gc Rank 3 : java.lang.Object.finalize Rank 4 : java.util.stream.Collectors.toList Rank 5 : java.util.Arrays.asList
How to get integer input in java?	Rank 1 : java.lang.Integer.valueOf Rank 2 : java.lang.Integer.parseInt Rank 3 : java.util.Scanner.nextInt Rank 4 : java.lang.Long.valueOf Rank 5 : java.lang.Byte.parseByte
how to print stacktrace in java?	Rank 1 : java.lang.Throwable.printStackTrace Rank 2 : java.lang.Thread.dumpStack Rank 3 : java.lang.Throwable.getStackTrace Rank 4 : java.lang.Throwable.getMessage Rank 5 : java.lang.Runtime.traceMethodCalls

Detailed Output:

QUERY _ 1 :

```
Enter the query:

How to get input from console in java?


Query as tokens: ['how', 'to', 'get', 'input', 'from', 'console', 'in',
'java']


Query after stemming: ['how', 'to', 'get', 'input', 'from', 'consol',
'in', 'java']


The query idf vectors for each word:


how ---> 2.533656505487701

to ---> 0.36778105765469626

get ---> 1.9413704063913035

input ---> 3.6045849339417164

from ---> 1.649573942917482

consol ---> 4.953771827154915

in ---> 0.6800034209749307

java ---> 1.7135938501242525


The Top 10 most relevant questions with their similarity to
query calculated using TF-IDF score:

// Output format (Question ID, Questions, TF-IDF score)
```

```
('41345374', 'How to take String input from console in Java?',
0.886887202053186)

('13023285', 'Java: How to get values from console input easily',
0.8655396973236977)

('11871520', 'How can I read input from the console using the
Scanner class in Java?', 0.8028075298426406)

('26684208', 'How to have multiple commands in a Java input
console', 0.7997196606712532)

('4906248', 'Get console text in java', 0.7897387208679262)

('19821794', 'How to use input obtained from a string in Java?',
0.7790072508026864)

('32339319', 'How to get Integer input from Java Scanner',
0.7775889412179967)

('5287538', 'How can I get the user input in Java?',
0.760350764831441)

('27692040', 'How to use a For loop to get user input in Java?',
0.753652348274386)
```

```
('19293093', 'how to get input stream from input stream file?',
0.7533825915058056)

Rank 1 : java.util.Scanner.next
Rank 2 : java.util.Scanner.nextLine
Rank 3 : java.lang.System.console
Rank 4 : java.io.BufferedReader.readLine
Rank 5 : java.io.FilterOutputStream.write

QUERY – 2:
Enter the query:
How to convert string to integer?

Query as tokens: ['how', 'to', 'convert', 'string', 'to', 'integer']

Query after stemming: ['how', 'to', 'convert', 'string', 'to', 'integ']

The query idf vectors for each word:
how ---> 2.533656505487701
to ---> 0.36778105765469626
convert ---> 3.692198693789381
string ---> 2.3297790033454473
to ---> 0.36778105765469626
integ ---> 3.7204130913957476
```

The Top 10 most relevant questions with their similarity to query calculated using TF-IDF score:

```
// Output format (Question ID, Questions, TF-IDF score)

 ('15017220', 'how to convert from string to integer',
0.95026020109232)

('3571352', 'How to convert Integer to int?', 0.9487721957309072)

('32907303', 'How to convert a string to an integer',
0.942150781988784)

('18294888', 'how to convert char to string',
0.9150377500696918)

('17214454', 'How to convert a hex integer to string',
0.9097358858783499)

('4841559', 'How to Convert an int to a String?',
0.8993497165165524)

('34387199', 'how to convert string of integers and Lists to
arrayList', 0.8990705061403388)

('32298126', 'How to convert the time to string or integer',
0.8858399454063901)

('28609364', 'How to convert int array to hex string',
0.8714552963107863)

('21800399', 'Trying to convert String to Integer',
0.8691598418530582)


Rank 1 : java.lang.String.toCharArray

Rank 2 : java.lang.Integer.toString
Rank 3 : java.lang.Integer.valueOf

Rank 4 : java.lang.Double.toString

Rank 5 : java.lang.Integer.intValue
```

QUERY – 3:

```
Enter the query:

How to convert integer to string?

Query as tokens: ['how', 'to', 'convert', 'integer', 'to', 'string']

Query after stemming: ['how', 'to', 'convert', 'integ', 'to', 'string']

The query idf vectors for each word:

how ---> 2.533656505487701

to ---> 0.36778105765469626

convert ---> 3.692198693789381

integ ---> 3.7204130913957476

to ---> 0.36778105765469626

string ---> 2.3297790033454473

The Top 10 most relevant questions with their similarity to
query calculated using TF-IDF score:

// Output format (Question ID, Questions, TF-IDF score)

 ('15017220', 'how to convert from string to integer',
0.95026020109232)

('3571352', 'How to convert Integer to int?', 0.9487721957309072)

('32907303', 'How to convert a string to an integer',
0.942150781988784)
```

```
('18294888', 'how to convert char to string',
0.9150377500696918)

('17214454', 'How to convert a hex integer to string',
0.9097358858783499)

('4841559', 'How to Convert an int to a String?',
0.8993497165165525)

('34387199', 'how to convert string of integers and Lists to
arrayList', 0.8990705061403388)

('32298126', 'How to convert the time to string or integer',
0.8858399454063901)

('28609364', 'How to convert int array to hex string',
0.8714552963107866)

('21800399', 'Trying to convert String to Integer',
0.8691598418530582)


Rank 1 : java.lang.String.toCharArray

Rank 2 : java.lang.Integer.toString

Rank 3 : java.lang.Integer.valueOf

Rank 4 : java.lang.Double.toString

Rank 5 : java.lang.Integer.intValue
```