NAME: NADAR KIBSON SAMUEL P JOSEPH

ROLL NO.: TCS2324045

SUBJECT: INFORMATION RETREIVAL JOURNAL

DEPARTMENT: TY BSC CS

SIES College of Arts, Science & Commerce

RISE WITH EDUCATION

NAAC REACCREDITED - 'A' GRADE

ISO 9001 : 2008

S.I.E.S College of Arts, Science and Commerce(Autonomous)
Sion(W), Mumbai – 400 022.

CERTIFICATE

This is to certify that Mr **NADAR KIBSON SAMUEL P JOSEPH** Roll No. **TCS2324045** has successfully completed the necessary course of experiments in the subject of **Information Retrieval** during the academic year **2023 – 2024** complying with the requirements of **University of Mumbai**, for the course of **TYBSc Computer Science [Semester-VI].**

Prof. In-Charge
Ramesh Ramnaresh Yadav

Examination date:

Examiner's Signature & Date:

Head of the Department
Prof. Manoj Singh

College Seal

Index Page

(Information Retrieval)
Practical No.1

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Bitwise Operation	**Batch**	I
Date:	21-12-23	Practical No	1

A) AIM: Write a python program to demonstrate bitwise operation

B) DESCRIPTION:

a) **& (AND Operator):** It takes two numbers as operands and does AND on every bit of two numbers. The result of AND is 1 only if both bits are 1.

b) **| (OR Operator):** It take two numbers as operands and does OR on every bit of two numbers. The result of OR is 1 if any of the two bits is 1.

c) **^ (XOR Operator):** It takes two numbers as operands and does XOR on every bit of two numbers. The result of XOR is 1 if the two bits are different.

d) **<< (Left shift Operator):** It takes two numbers, the left shifts the bits of the first operand, and the second operand decides the number of places to shift.

e) **>> (Right Shift Operator):** It takes two numbers, right shifts the bits of the first operand, and the second operand decides the number of places to shift.

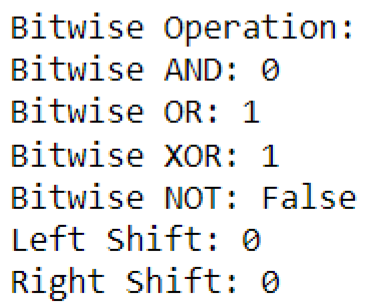
f) **~ (bitwise NOT):** It takes one number and inverts all bits of it.

C) CODE :

```
# Write a python program to demonstrate bitwise operation
```

```
def bitwise_operation(a,b):

   # AND
   bitwise_and_result = a & b
   print(f"Bitwise AND: {bitwise_and_result}")

   # OR
   bitwise_or_result = a | b
   print(f"Bitwise OR: {bitwise_or_result}")



   # XOR
   bitwise_xor_result = a ^ b
   print(f"Bitwise XOR: {bitwise_xor_result}")

   # NOT
   bitwise_not_result = a == ~b
   print(f"Bitwise NOT: {bitwise_not_result}")

   # Left shift
   left_shift = a << 1
   print(f"Left Shift: {left_shift}")

   # Right shift
   right_shift = a >> 1
   print(f"Right Shift: {right_shift}")

   return None

print("Bitwise Operation: ")



bitwise_operation(0,1)
```

OUTPUT:



```
Bitwise Operation:
Bitwise AND: 0
Bitwise OR: 1
Bitwise XOR: 1
Bitwise NOT: False
Left Shift: 0
Right Shift: 0
```

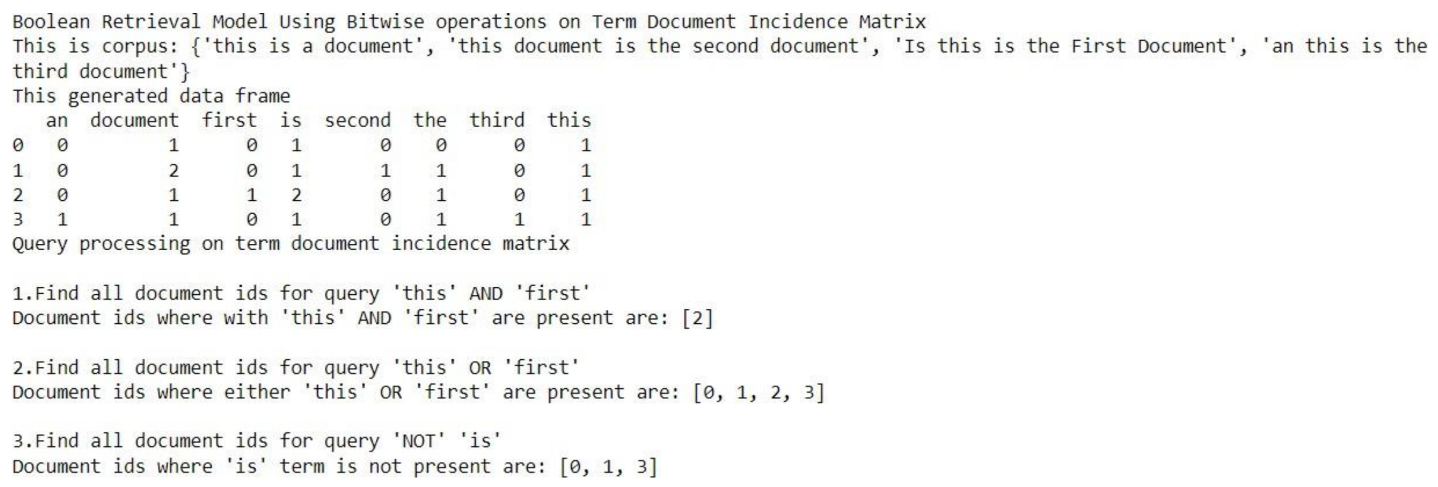
Method 2:

INPUT:

```
# Practical No 1

# Method 2

# For Text


import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer

print("Boolean Retrieval Model Using Bitwise operations on Term Document Incidence Matrix")


corpus = {'this is a document', 'this document is the second document',

          'an this is the third document', 'Is this is the First Document'}




# type(corpus)

print(f"This is corpus: {corpus}")

vectorizer = CountVectorizer()

x = vectorizer.fit_transform(corpus)

df = pd.DataFrame(x.toarray(),columns=vectorizer.get_feature_names_out())

print("This generated data frame")

print(df)

print("Query processing on term document incidence matrix \n")
```

```
# AND

print("1.Find all document ids for query 'this' AND 'first'")

alldata = df[(df['this'] == 1) & (df['first'] == 1)]

print(f"Document ids where with 'this' AND 'first' are present are: {alldata.index.tolist()} \n")



# OR

print("2.Find all document ids for query 'this' OR 'first'")

alldata = df[(df['this'] == 1) | (df['first'] == 1)]

print(f"Document ids where either 'this' OR 'first' are present are: {alldata.index.tolist()} \n")



# NOT

print("3.Find all document ids for query 'NOT' 'is'")

alldata = df[(df['is'] == 1)]

print(f"Document ids where 'is' term is not present are: {alldata.index.tolist()} \n")
```

OUTPUT:



```
Boolean Retrieval Model Using Bitwise operations on Term Document Incidence Matrix
This is corpus: {'this is a document', 'this document is the second document', 'Is this is the First Document', 'an this is the
third document'}
This generated data frame
   an  document  first  is  second  the  third  this
0   0         1      0   1       0    0      0     1
1   0         2      0   1       1    1      0     1
2   0         1      1   2       0    1      0     1
3   1         1      0   1       0    1      1     1
Query processing on term document incidence matrix

1.Find all document ids for query 'this' AND 'first'
Document ids where with 'this' AND 'first' are present are: [2]

2.Find all document ids for query 'this' OR 'first'
Document ids where either 'this' OR 'first' are present are: [0, 1, 2, 3]

3.Find all document ids for query 'NOT' 'is'
Document ids where 'is' term is not present are: [0, 1, 3]
```

(InformationRetrieval)

Practical No.2

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:		**Batch**	I
Date:	03-01-2024	Practical No	2

A) AIM: Write a python code for implementation PageRank using NetworkX.

B) DESCRIPTION:

Implementation of PageRank Using NetworkX:

About NetworksX : NetoworkX is a Python package for the Creation ,Manipulation of the Structure , Dynamics and Function of Complex Networks.

About Pylab: Pylab is a convenience module that bulk imports matplotlib.pyplot(for Plotting) And Numpy(for Mathematics and Working with Arrays) in a Single name space Although many examples use Pylab

Although many Examples use Pylab it is no longer recommended , Installation The Pylab Module at the Matplotlib Package

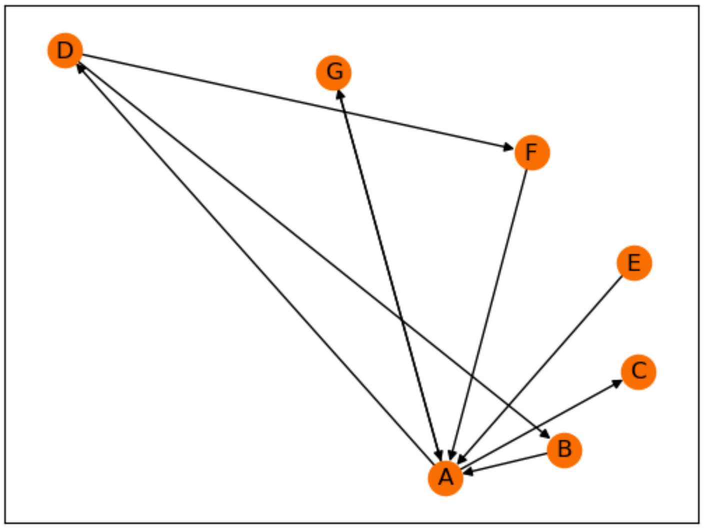
C) CODE :

1.Without Weighted Egdes

```
import networkx as nx
import pylab as plt
```

```
G=nx.DiGraph()
[G.add_node(k) for k in["A","B","C","E","F","G"]]
G.add_edges_from([("A","G"),("G","A"),("F","A"),("E","A"),("A","D"),("A","C"),("D","B"),
                ("B","A"),("A","C"),("D","F")])

ppr1=nx.pagerank(G)
print("Page rank value :",ppr1)
pos=nx.spiral_layout(G)
nx.draw_networkx(G,pos,with_labels = True,node_color="#f86e00")
plt.show()
```

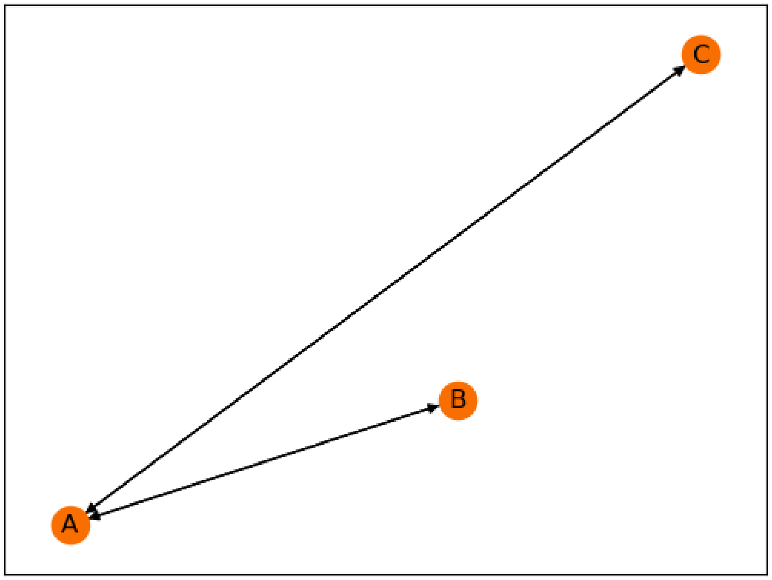
```
Page rank value : {'A': 0.35307327731863686, 'B': 0.09697413777431879, 'C': 0.1382539906905885, 'E': 0.03821647506095973,
'F': 0.09697413777431879, 'G': 0.1382539906905885, 'D': 0.1382539906905885}
```



2. With Weighted Edges

```
import networkx as nx
import pylab as plt
D=nx.DiGraph()
D.add_weighted_edges_from([('A','B','1'),('A','C','1'),('B','A','1'),('C','A',1)])
ppr1=nx.pagerank(D)
print("Page rank value :",ppr1)
nx.draw_networkx(D,pos,with_labels = True,node_color="#f86e00")
plt.show()
```

```
Page rank value : {'A': 0.48648582432442095, 'B': 0.25675708783778944, 'C': 0.25675708783778944}
```

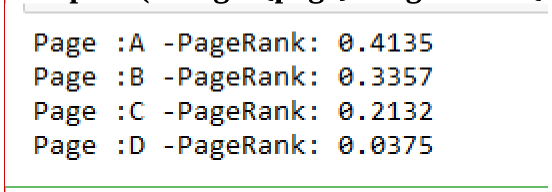


3.Method 3

```
def page_rank(graph, damping_factor=0.85, max_iterations=100, tolerance=1e-6):
    num_pages = len(graph)
    initial_page_rank = 1 / num_pages

    page_ranks = {page: initial_page_rank for page in graph}

    for _ in range(max_iterations):
        new_page_ranks = {}

        for page in graph:
            new_rank = (1 - damping_factor) / num_pages

            for link in graph:
                if page in graph[link]:
                    new_rank += damping_factor * (page_ranks[link] / len(graph[link]))

            new_page_ranks[page] = new_rank

        convergence = all(abs(new_page_ranks[page] - page_ranks[page]) < tolerance for page in graph)
        page_ranks = new_page_ranks

        if convergence:
            break

    return page_ranks

if__name__== "_main_":
    example_graph = {
        'A': ['B', 'C'],
```

Name of Instructor: Mr.RajeshYadav

```
    'B': ['A'],
    'C': ['A', 'B'],
    'D': ['B']
}

result = page_rank(example_graph)
for page,rank in sorted(result.items(),key=lambda x:x[1],reverse=True):
    print(f"Page :{page} -PageRank: {rank:.4f}")
```



```
Page :A -PageRank: 0.4135
Page :B -PageRank: 0.3357
Page :C -PageRank: 0.2132
Page :D -PageRank: 0.0375
```

(Information Retrieval)

Practical No.3

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Levenshtein Distance	**Batch**	I
Date:	09-01-2024	Practical No	3

A) AIM: Write a program to implement Levenshtein Distance.

B) DESCRIPTION:

Levenshtein distance is a measure of the similarity between two strings, which takes into account the number of insertion, deletion and substitution operations needed to transform one string into the other.

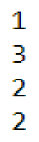
Operations in Levenshtein distance are:

- **Insertion:** Adding a character to string A.
- **Deletion:** Removing a character from string A.
- **Replacement:** Replacing a character in string A with another character.

C) CODE AND OUTPUT:

```
#levensien distance

def leven(x,y):

    n=len(x)

    m=len(y)
    A=[[i+j for j in range(m+1)] for i in range(n+1)]
    for i in range(n):
        for j in range (m):
            A[i+1][j+1]=min(A[i][j+1]+1,
                            A[i+1][j]+1,
                            A[i][j]+int(x[i]!=y[j]))
    return A[n][m]
print(leven("brap","rap"))
print(leven("trial","try"))
print(leven("horse","force"))
print(leven("rose","erode"))
```

Output:



```
1
3
2
2
```

(Information Retrieval)

Practical No.4

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Similarity between two documents	**Batch**	I
Date:	09-01-2024	Practical No	4

A) AIM: Write a program to Compute Similarity between two text documents.

B) DESCRIPTION:

Jaccard Similarity:

Jaccard Similarity is calculated as the size of the intersection of sets divided by the size of the union of sets.

J(A,B)=|\frac{|A\cap B|}{A||B|}

Cosine Similarity:

Cosine Similarity is calculated as the dot product of the vectors representing the sets divided by the product of their magnitudes.

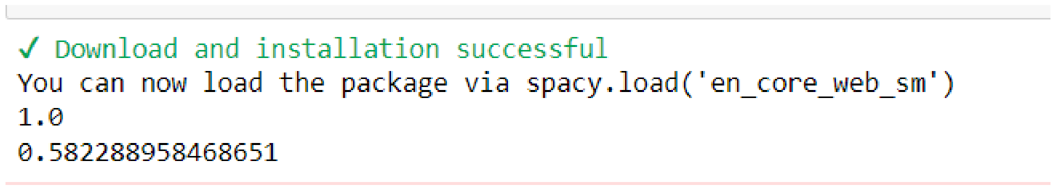
Cosine Similarity (A,B)=\frac{A\cdot B}{||A|\cdot|B||}

C) CODE AND OUTPUT:

METHOD 1:

```
import spacy

from spacy.cli.download import download

download(model="en_core_web_sm")

nlp=spacy.load('en_core_web_sm')

doc1=nlp(u'Hello hi there!')

doc2=nlp(u'Hello hi there!')

doc3=nlp(u'Hey whatsup?')

print(doc1.similarity(doc2))

print(doc1.similarity(doc3))
```

OUTPUT :

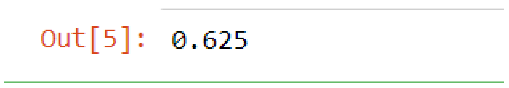


```
✓ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
1.0
0.582288958468651
```

METHOD 2: JACCARD SIMILARITY

```
def Jacard_Similarity(doc1,doc2):

   words_doc1=set(doc1.lower().split())

   words_doc2=set(doc2.lower().split())

   intersection=words_doc1.intersection(words_doc2)

   union=words_doc1.union(words_doc2)

   return float(len(intersection))/len(union)

doc1="data is the new oil of the digital economy"

doc2="data is the new oil "

Jacard_Similarity(doc1,doc2)
```

OUTPUT :

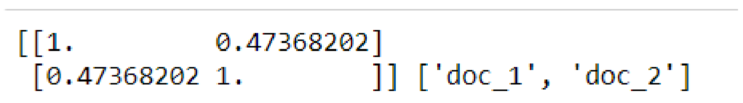


```
Out[5]: 0.625
```

METHOD 3: COSINE SIMILARITY

```
from sklearn.metrics.pairwise import cosine_similarity

doc_1="Data is the new oil of the digital economy"

doc_2="Data is a new oil"

data=[doc_1,doc_2]

from sklearn.feature_extraction.text import TfidfVectorizer

Tfidf_vect=TfidfVectorizer()

vector_matrix=Tfidf_vect.fit_transform(data)

tokens=Tfidf_vect.get_feature_names_out()

create_dataframe=(vector_matrix.toarray(),tokens)

cosine_similarity_matrix=cosine_similarity(vector_matrix)

create_dataframe=print(cosine_similarity_matrix,['doc_1','doc_2'])
```

OUTPUT :



```
[[1.         0.47368202]
 [0.47368202 1.        ]] ['doc_1', 'doc_2']
```

Name:	Nadar Kibson Samuel P Joseph	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Map reducer	**Batch**	I
Date:	24-01-2024	Practical No	5

A) AIM: Write a python program for map reducer.

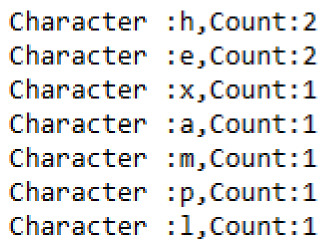
B) DESCRIPTION:

Map-Reduce is a programming model that is mainly divided into two phases i.e. Map Phase and Reduce Phase. It is designed for processing the data in parallel which is divided on various machines (nodes). The Mapper produces the output in the form of key-value pairs which works as input for the Reducer.

C) CODE AND OUTPUT:

```
from functools import reduce
from collections import defaultdict
def mapper(data):
    char_count=defaultdict(int)
    for char in data:
        if char.isalpha():
            char_count[char.lower()]+=1
    return char_count.items()
def reducer (counts1,counts2):
    merged_counts=defaultdict(int)
    for char,count in counts1:
        merged_counts[char]+=count
        for char,count in counts2:
            merged_counts[char]+=count
        return merged_counts.items()
if__name__=="__main__":
    dataset="Hello,World! This is a MapReducer example"
    #split the dataset into chunks(assuming a distributed env)
chunks=[chunk for chunk in dataset.split()]
    #Map step
    mapped_results=map(mapper,chunks)
    #Reduce step
    final_counts=reduce(reducer,mapped_results)
    #Print the result
    for char,count in final_counts:
        print(f"Character :{char},Count:{count}")
```

OUTPUT:



```
Character :h,Count:2
Character :e,Count:2
Character :x,Count:1
Character :a,Count:1
Character :m,Count:1
Character :p,Count:1
Character :l,Count:1
```

(Information Retrieval)

Practical No.6

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:		**Batch**	I
Date:	24/01/2024	Practical No	6

A) AIM: Write a python program for HITS Algorithm

C) DESCRIPTION:

Hyperlink-Induced Topic Search (HITS; also known as hubs and authorities) is a link analysis algorithm that rates Web pages. The idea behind Hubs and Authorities stemmed from a particular insight into the creation of web pages when the Internet was originally forming; that is, certain web pages, known as hubs, served as large directories that were not actually authoritative in the information that they held, but were used as compilations of a broad catalog of information that led users direct to other authoritative pages.

C) CODE AND OUTPUT:

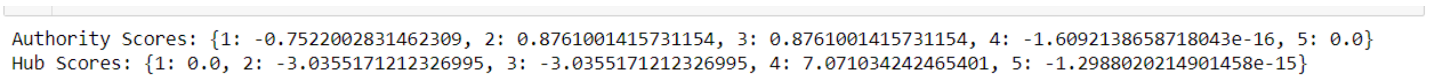
METHOD 7A:

CODE:

```
#HITS Algorithm

import networkx as nx

# Step 2: Create a graph and add edges

G = nx.DiGraph()

G.add_edges_from([(1, 2), (1, 3), (2, 4), (3, 4), (4, 5)])


# Step 3: Calculate the HITS scores
```

```
authority_scores, hub_scores = nx.hits(G)


# Step 4: Print the scores

print("Authority Scores:", authority_scores)

print("Hub Scores:",hub_scores)
```

OUTPUT:



```
Authority Scores: {1: -0.7522002831462309, 2: 0.8761001415731154, 3: 0.8761001415731154, 4: -1.6092138658718043e-16, 5: 0.0}
Hub Scores: {1: 0.0, 2: -3.0355171212326995, 3: -3.0355171212326995, 4: 7.071034242465401, 5: -1.2988020214901458e-15}
```

(Information Retrieval)
Practical No.7

DEPARTMENT OF COMPUTER SCIENCE

Name:	Nadar Kibson Samuel P Joseph	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Page Rank	**Batch**	I
Date:	24-01-2024	Practical No	7

A) AIM: Write a python program for Stop Words.

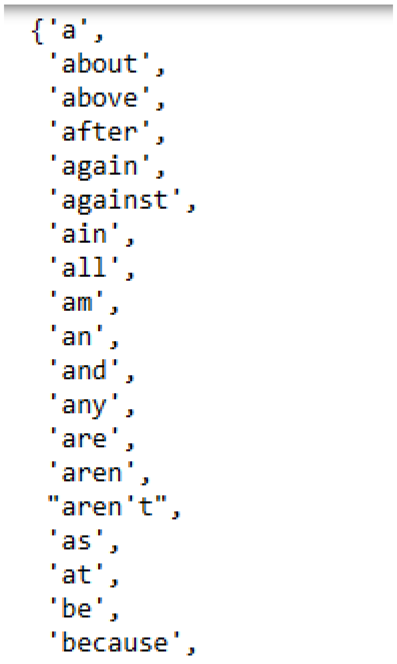
B) DESCRIPTION:
Stop words are commonly used words excluded from searches to help index and parse web pages faster. While most Internet search engines and NLP (natural language processing) utilize stop words, they do not prevent users from using them. Instead, the words are only ignored when the search results are displayed.

C) CODE AND OUTPUT:
METHOD 7A:
CODE:

```
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
set(stopwords.words('english'))
```

Name of Instructor: Mr.RajeshYadav

OUTPUT:



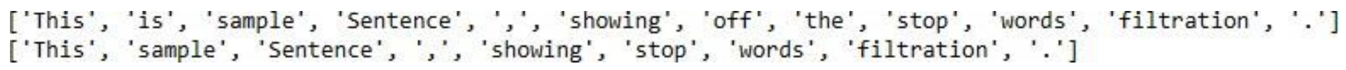
```
{'a',
 'about',
 'above',
 'after',
 'again',
 'against',
 'ain',
 'all',
 'am',
 'an',
 'and',
 'any',
 'are',
 'aren',
 "aren't",
 'as',
 'at',
 'be',
 'because',
```

METHOD 7B:

CODE:

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

example_sent = "This is sample Sentence , showing off the stop words filtration."
stop_words=set(stopwords.words('english'))
word_tokens=word_tokenize(example_sent)
filtered_sentence=[w for w in word_tokens if not w in stop_words]
filtered_sentence=[]

for w in word_tokens:
    if w not in stop_words:
        filtered_sentence.append(w)

print(word_tokens)
print(filtered_sentence)
```

OUTPUT:



```
['This', 'is', 'sample', 'Sentence', ',', 'showing', 'off', 'the', 'stop', 'words', 'filtration', '.']
['This', 'sample', 'Sentence', ',', 'showing', 'stop', 'words', 'filtration', '.']
```

(Information Retrieval)
Practical No.8

DEPARTMENT OF COMPUTER SCIENCE

Name:	Nadar Kibson Samuel P Joseph	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Twitter Scraping	**Batch**	I
Date:	24-01-2024	Practical No	8

A) AIM: Write a python program for Twitter Scraping using Nitter.

B) DESCRIPTION:

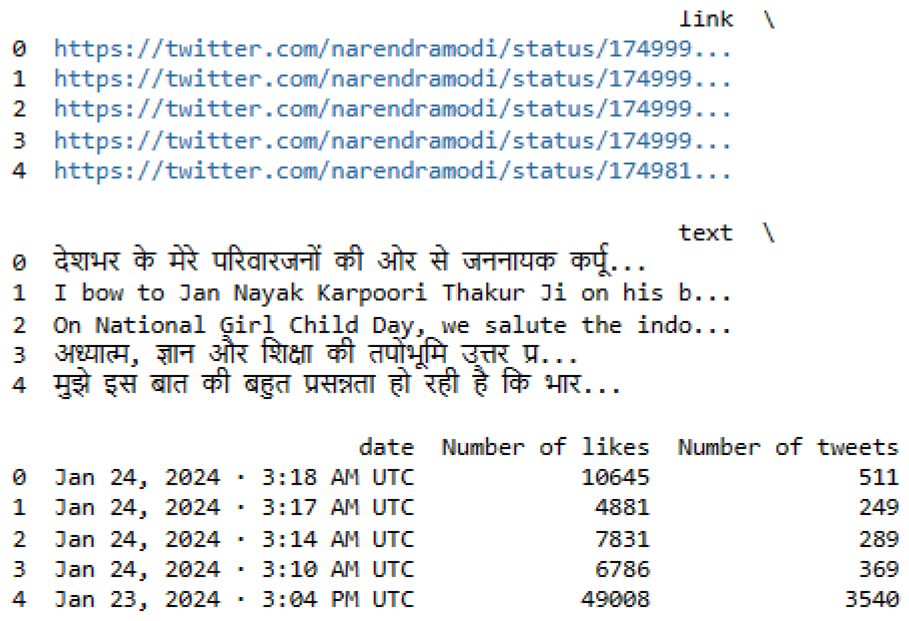
This is a simple library to scrape Nitter instances for tweets. It can: search and scrape tweets with a certain term search and scrape tweets with a certain hashtag scrape tweets from a user profile get profile information of a user, such as display name, username, number of tweets, profile picture.

C) CODE AND OUTPUT:

CODE:

```
!pip install ntscraper
import pandas as pd
from ntscraper import Nitter
scraper=Nitter()
tweets=scraper.get_tweets('narendramodi',mode='user',number=5)
final_tweets=[]
for tweet in tweets['tweets']:
data=[tweet['link'],tweet['text'],tweet['date'],tweet['stats']['likes'],tweet['stats']['comments']]
    final_tweets.append(data)
print(final_tweets)
data=pd.DataFrame(final_tweets,columns=['link','text','date','Number of likes','Number of tweets'])
print(data)
```

OUTPUT:



```
                                                link  \
0  https://twitter.com/narendramodi/status/174999...
1  https://twitter.com/narendramodi/status/174999...
2  https://twitter.com/narendramodi/status/174999...
3  https://twitter.com/narendramodi/status/174999...
4  https://twitter.com/narendramodi/status/174981...

                                                text  \
0  देशभर के मेरे परिवारजनों की ओर से जननायक कर्पू...
1  I bow to Jan Nayak Karpoori Thakur Ji on his b...
2  On National Girl Child Day, we salute the indo...
3  अध्यात्म, ज्ञान और शिक्षा की तपोभूमि उत्तर प्र...
4  मुझे इस बात की बहुत प्रसन्नता हो रही है कि भार...

                        date  Number of likes  Number of tweets
0  Jan 24, 2024 · 3:18 AM UTC            10645               511
1  Jan 24, 2024 · 3:17 AM UTC             4881               249
2  Jan 24, 2024 · 3:14 AM UTC             7831               289
3  Jan 24, 2024 · 3:10 AM UTC             6786               369
4  Jan 23, 2024 · 3:04 PM UTC            49008              3540
```

SIES

(Information Retrieval)

Practical No.9

DEPARTMENT OF COMPUTER SCIENCE

Name:	Nadar Kibson Samuel P Joseph	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	Basic Web crawler	**Batch**	I
Date :	29/01/24	Practical No	9

A) AIM:

Write a python program to implement simple web crawling.

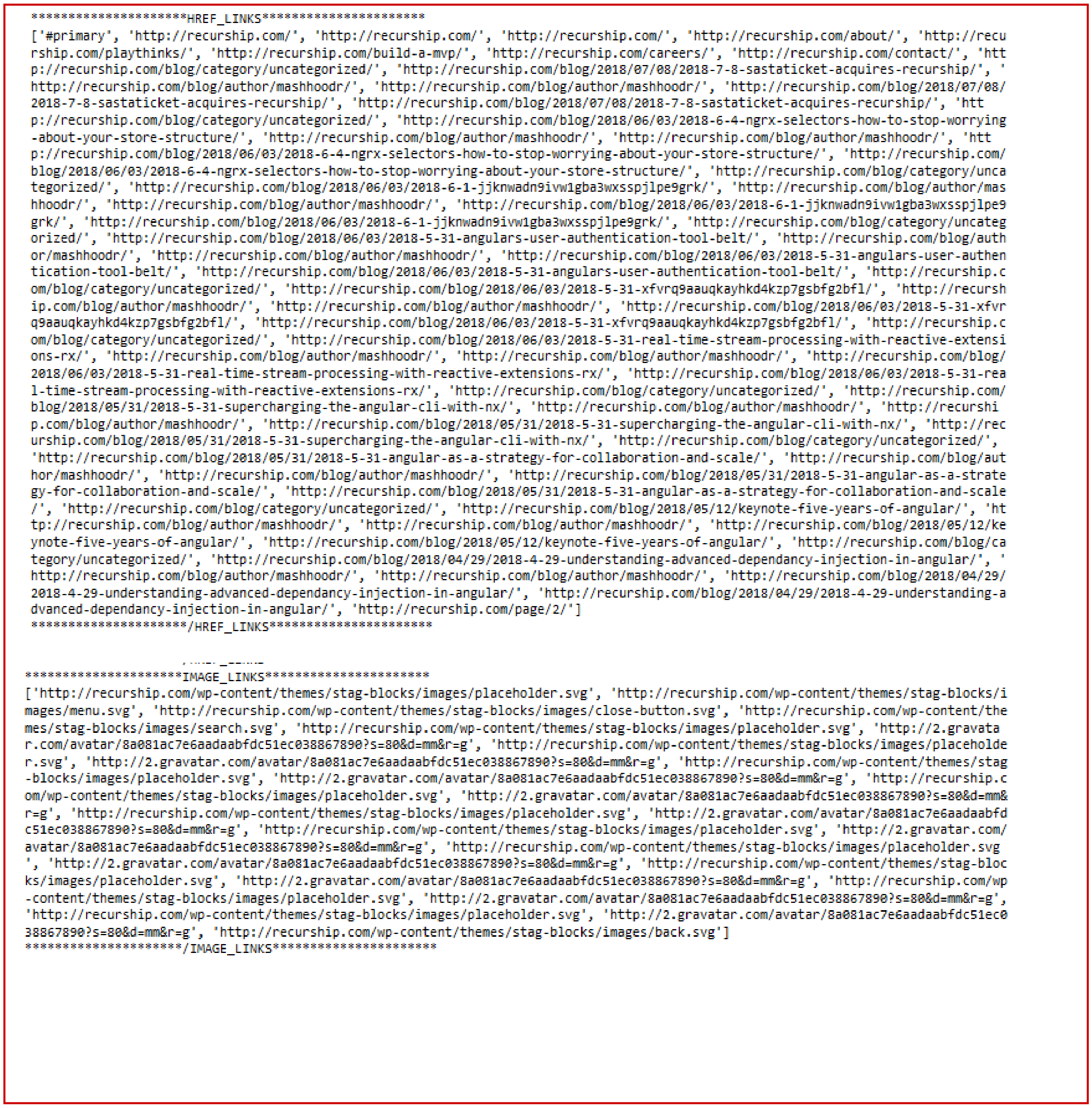
B) DESCRIPTION:

Web Crawling is the process by which we gather pages from the web in order to index them and support the search engine.

- The crawler begins with one or more URLs that constitute a set seed.
- It picks a URL from the seed set, and then fetches the web page at that URL
- The fetched page is then parsed, to extract both text and the links from the page(each of which points to another URL)
- The extracted URL is fed to a text indexer
- The extracted URLs are then added to a URL frontier, which at all times consists of URLs who's corresponding pages have yet to be fetched by the crawler.
- Initially, the URL frontier contains the seed set as pages are fetched the corresponding URLs are deleted from the URL frontier. The entire process can be viewed as traversing the web graph. In continuous crawling the URL of the fetched page is added back to the URL frontier.

C) CODE AND OUTPUT:

```
#PRactical 9

import requests

from parsel import Selector

import time


start=time.time

response = requests.get('http://recurship.com/')

selector=Selector(response.text)

href_links=selector.xpath('//a/@href').getall()

image_links=selector.xpath('//img@src').getall()

print('*********************HREF_LINKS**********************')

print(href_links)

print('*********************/HREF_LINKS**********************')


#datatype of this is Lists


print('*********************IMAGE_LINKS**********************')

print(image_links)

print('*********************/IMAGE_LINKS**********************')
```



```
*********************HREF_LINKS**********************
['#primary', 'http://recurship.com/', 'http://recurship.com/', 'http://recurship.com/', 'http://recurship.com/about/', 'http://recu
rship.com/playthinks/', 'http://recurship.com/build-a-mvp/', 'http://recurship.com/careers/', 'http://recurship.com/contact/', 'htt
p://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/07/08/2018-7-8-sastaticket-acquires-recurship/', '
http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/07/08/
2018-7-8-sastaticket-acquires-recurship/', 'http://recurship.com/blog/2018/07/08/2018-7-8-sastaticket-acquires-recurship/', 'htt
p://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/06/03/2018-6-4-ngrx-selectors-how-to-stop-worrying
-about-your-store-structure/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'htt
p://recurship.com/blog/2018/06/03/2018-6-4-ngrx-selectors-how-to-stop-worrying-about-your-store-structure/', 'http://recurship.com/
blog/2018/06/03/2018-6-4-ngrx-selectors-how-to-stop-worrying-about-your-store-structure/', 'http://recurship.com/blog/category/unca
tegorized/', 'http://recurship.com/blog/2018/06/03/2018-6-1-jjknwadn9ivw1gba3wxsspjlpe9grk/', 'http://recurship.com/blog/author/mas
hhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/06/03/2018-6-1-jjknwadn9ivw1gba3wxsspjlpe9
grk/', 'http://recurship.com/blog/2018/06/03/2018-6-1-jjknwadn9ivw1gba3wxsspjlpe9grk/', 'http://recurship.com/blog/category/uncateg
orized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authentication-tool-belt/', 'http://recurship.com/blog/auth
or/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authen
tication-tool-belt/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authentication-tool-belt/', 'http://recurship.c
om/blog/category/uncategorized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recursh
ip.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvr
q9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.c
om/blog/category/uncategorized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-real-time-stream-processing-with-reactive-extensi
ons-rx/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/
2018/06/03/2018-5-31-real-time-stream-processing-with-reactive-extensions-rx/', 'http://recurship.com/blog/2018/06/03/2018-5-31-rea
l-time-stream-processing-with-reactive-extensions-rx/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/
blog/2018/05/31/2018-5-31-supercharging-the-angular-cli-with-nx/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurshi
p.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/05/31/2018-5-31-supercharging-the-angular-cli-with-nx/', 'http://rec
urship.com/blog/2018/05/31/2018-5-31-supercharging-the-angular-cli-with-nx/', 'http://recurship.com/blog/category/uncategorized/',
'http://recurship.com/blog/2018/05/31/2018-5-31-angular-as-a-strategy-for-collaboration-and-scale/', 'http://recurship.com/blog/aut
hor/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/05/31/2018-5-31-angular-as-a-strate
gy-for-collaboration-and-scale/', 'http://recurship.com/blog/2018/05/31/2018-5-31-angular-as-a-strategy-for-collaboration-and-scale
/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/05/12/keynote-five-years-of-angular/', 'ht
tp://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/05/12/ke
ynote-five-years-of-angular/', 'http://recurship.com/blog/2018/05/12/keynote-five-years-of-angular/', 'http://recurship.com/blog/ca
tegory/uncategorized/', 'http://recurship.com/blog/2018/04/29/2018-4-29-understanding-advanced-dependancy-injection-in-angular/', '
http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/04/29/
2018-4-29-understanding-advanced-dependancy-injection-in-angular/', 'http://recurship.com/blog/2018/04/29/2018-4-29-understanding-a
dvanced-dependancy-injection-in-angular/', 'http://recurship.com/page/2/']
*********************/HREF_LINKS**********************


*********************IMAGE_LINKS**********************
['http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://recurship.com/wp-content/themes/stag-blocks/i
mages/menu.svg', 'http://recurship.com/wp-content/themes/stag-blocks/images/close-button.svg', 'http://recurship.com/wp-content/the
mes/stag-blocks/images/search.svg', 'http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://2.gravata
r.com/avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag-blocks/images/placeholde
r.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag
-blocks/images/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&r=g', 'http://recurship.c
om/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&
r=g', 'http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfd
c51ec038867890?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://2.gravatar.com/
avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg
', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag-bloc
ks/images/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&r=g', 'http://recurship.com/wp
-content/themes/stag-blocks/images/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec038867890?s=80&d=mm&r=g',
'http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec0
38867890?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag-blocks/images/back.svg']
*********************/IMAGE_LINKS**********************
```

Information Retrieval

Practical No.10

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:		**Batch**	I
Date :	07/01/24	Practical No	10

A) AIM:

to demonstrate the parsing of an XML document representing a web graph, generating a directed graph (DiGraph) using the NetworkX library, and computing the topic-specific PageRank for specific nodes within the graph.

B) DESCRIPTION:

Parsing XML:

The code starts by defining a function parse_xml to parse an XML document using the ElementTree module.

Generating Web Graph:

Another function generate_web_graph is defined to take the parsed XML root and create a directed graph (DiGraph) using the NetworkX library.

It iterates through each <page> element, extracts the page ID and links, and adds nodes and edges to the graph accordingly.

Computing Topic-Specific PageRank:

The compute_topic_specific_pagerank function calculates the PageRank of nodes in the graph, considering a topic-specific personalization vector.

It uses NetworkX's pagerank function, providing a personalization dictionary where nodes in the specified topic_nodes receive a higher initial weight.

Example Usage:

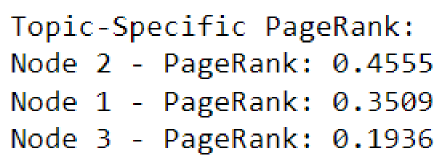
In the __main__block, an example XML text is provided, representing web pages and links.

The XML is parsed, a web graph is generated, and topic-specific PageRank is computed for nodes "1" and "2".

The results are then printed, showing the PageRank scores for the specified nodes in descending order.

Overall, this code serves as a basic example of utilizing XML parsing and network analysis techniques to compute topic-specific PageRank in a web graph.

```
import xml.etree.ElementTree as ET

import networkx as nx


def parse_xml(xml_text):

    root = ET.fromstring(xml_text)

    return root


def generate_web_graph(xml_root):

    G = nx.DiGraph()

    for page in xml_root.findall(".//page"):

        page_id = page.find("id").text

        G.add_node(page_id)

        links = page.findall(".//link")

        for link in links:

            target_page_id = link.text
            G.add_edge(page_id, target_page_id)

    return G


def compute_topic_specific_pagerank(graph, topic_nodes, alpha=0.85, max_iter=100, tol=1e-6):

    personalization = {node: 1.0 if node in topic_nodes else 0.0 for node in graph.nodes}

    return nx.pagerank(graph, alpha=alpha, personalization=personalization,
max_iter=max_iter, tol=tol)


if __name__ == "__main__":

    # Example XML text representing web pages and links

    example_xml = '''

    <webgraph>

        <page>

            <id>1</id>

            <link>2</link>

            <link>2</link>

        </page>

        <page>

            <id>2</id>

            <link>1</link>

            <link>3</link>

        </page>

        <page>

            <id>3</id>

            <link>1</link>
        <link>2</link>
    </page>
</webgraph>
'''

# Parse XML
xml_root = parse_xml(example_xml)

# Generate web graph
web_graph = generate_web_graph(xml_root)

# Compute topic-specific PageRank for nodes 1 and 2
topic_specific_pagerank = compute_topic_specific_pagerank(web_graph, topic_nodes=["1", "2"])

# Print the result
print("Topic-Specific PageRank:")
for node, score in sorted(topic_specific_pagerank.items(), key=lambda x: x[1], reverse=True):
    print(f"Node {node} - PageRank: {score:.4f}")
```



```
Topic-Specific PageRank:
Node 2 - PageRank: 0.4555
Node 1 - PageRank: 0.3509
Node 3 - PageRank: 0.1936
```

SIES

Information Retrieval

Practical No.11

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:		**Batch**	I
Date :	07/02/2024	Practical No	11

A) AIM:

Write a python program to retrieve xml text using xml library.

B) DESCRIPTION:

An Extensible Markup Language (XML) file is a text-based document that you can save with the .xml extension. You can write XML similar to other text files. To create or edit an XML file, you can use any of the following: Text editors like Notepad or Notepad++ Online XML editors.

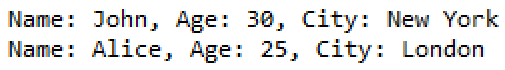
XML retrieval, or XML information retrieval, is the content-based retrieval of documents structured with XML (eXtensible Markup Language). As such it is used for computing relevance of XML documents.

C) CODE AND OUTPUT:

```
import xml.etree.

ElementTree as ET



xml_data = '''<root>

<person>
```

Name of Instructor: Mr.Rajesh Yadav

```
<name>John</name>

<age>30</age>

<city>New York</city>

</person>

<person>

<name>Alice</name>

<age>25</age>

<city>London</city>

</person>

</root>'''


tree = ET.fromstring(xml_data)


for person in tree.findall('person'): name = person.find('name').text age = person.find('age').text city =
person.find('city').text

print(f"Name: {name}, Age: {age}, City: {city}")
```



```
Name: John, Age: 30, City: New York
Name: Alice, Age: 25, City: London
```

SIES

Information Retrieval

Practical No.12

DEPARTMENT OF COMPUTER SCIENCE

Name:	NADAR KIBSON SAMUEL P JOSEPH	**Roll Number**	TCS2324045
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	lxml library	**Batch**	I
Date :	07/02/2024	Practical No	12

A) AIM:

Write a python program to retrieve xml text using lxml library.

B) DESCRIPTION:

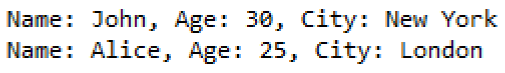
An Extensible Markup Language (XML) file is a text-based document that you can save with the .xml extension. You can write XML similar to other text files. To create or edit an XML file, you can use any of the following: Text editors like Notepad or Notepad++ Online XML editors.

XML retrieval, or XML information retrieval, is the content-based retrieval of documents structured with XML (eXtensible Markup Language). As such it is used for computing relevance of XML documents.

C) CODE AND OUTPUT:

```
import xml.etree.ElementTree as ET



xml_data = '''<root>

<person>

<name>John</name>
```

```
<age>30</age>

<city>New York</city>

</person>

<person>

<name>Alice</name>

<age>25</age>

<city>London</city>

</person>

</root>'''



tree = ET.fromstring(xml_data)



for person in tree.findall('person'): name = person.find('name').text age = person.find('age').text city =
person.find('city').text

print(f"Name: {name}, Age: {age}, City: {city}")
```



```
Name: John, Age: 30, City: New York
Name: Alice, Age: 25, City: London
```