College of Arts, Science & Commerce (Autonomous)

SIES

RISE WITH EDUCATION

NAAC REACCREDITED - 'A' GRADE

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

CERTIFICATE

This is to certify that Mr. /~~Miss~~. **Kamal Jitesh Vasa**

Roll No. **TCS2324087** 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 **T.Y. BSc. Computer Science [Semester-6]**

Prof. In-Charge
Prof. Rajesh Yadav
(Information Retrieval)

Examination Date:
Examiner's Signature & Date:

Head of the Department
Prof. Manoj Singh

College Seal
And
Date

INDEX PAGE				
Sr. No	Description	Page No	Date	Faculty Signature
1	**Write a python program to demonstrate bitwise operation.**	3	21-12-23	
2	**Implement Page Rank Algorithm**	7	3-1-24	
3	**Write a program to implement Levenshtein Distance.**	11	9-1-24	
4	**Write a program to Compute Similarity between two text documents.**	13	9-1-24	
5	**Write a Map Reduce Program to count the number of occurrences of each alphabetic character in a given dataset.**	16	20-1-24	
6	**HITS Algorithm**	19	20-1-24	
7	**Write a python program for pre-processing of text document:stopword removal.**	21	24-1-24	
8	**Write a program for mining twitter to identify tweets for a specific period and identify trends and named entities.**	24	24-1-24	
9	**Write a program to implement simple web crawling.**	27	29-1-24	
10	**Write a python program to parse XML text, generate Web graph and compute topic specific page rank.**	31	7-2-24	

Information Retrieval
Practical No.1

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal Vasa	**Roll Number**	TCS2324087
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.

METHOD I

B) DESCRIPTION:

Bitwise AND: The output of bitwise AND is **1** if the corresponding bits of two operands is **1**. If either bit of an operand is **0**, the result of corresponding bit is evaluated to **0**.
Bitwise OR: The output of bitwise OR is **1** if at least one corresponding bit of two operands is **1**. In C Programming, bitwise OR operator is denoted by |.
Bitwise XOR: The result of bitwise XOR operator is **1** if the corresponding bits of two operands are opposite. It is denoted by ^.
Bitwise NOT: Bitwise NOT operator works on only one operand. It changes **1** to **0** and **0** to **1**. It is denoted by ~.
Right shift: Right shift operator shifts all bits towards right by certain number of specified bits. It is denoted by >>.
Left shift: Left shift operator shifts all bits towards left by a certain number of specified bits. The bit positions that have been vacated by the left shift operator are filled with **0**. The symbol of the left shift operator is <<.

C) CODE AND OUTPUT:

METHOD 1:

```
def bitwise_operations(a,b):
    bitwise_and= a & b #bitwise AND operation
    print("bitwise AND",bitwise_and)
```

```
    bitwise_or= a | b #bitwise OR operation
    print(f"bitwise OR {bitwise_or}")
    bitwise_xor= a ^ b  #bitwise XOR operation
    print(f"bitwise XOR {bitwise_xor}")
    bitwise_not= ~a #bitwise NOT operation
    print(f"bitwise NOT of a {bitwise_not}")
    bitwise_not= ~b #bitwise NOT operation
    print(f"bitwise NOT of b {bitwise_not}")
    bitwise_left= a<<1 #bitwise LEFT-Shift operation
    print(f"bitwise LEFT {bitwise_left}")
    bitwise_right= a>>2 #bitwise RIGHT-shift operation
    print(f"bitwise RIGHT {bitwise_right}")
a=int(input("Enter the value of a: "))
b=int(input("Enter the value of a: "))
bitwise_operations(a,b)


Enter the value of a: 10110010
Enter the value of a: 0010
bitwise AND 10
bitwise OR 10110010
bitwise XOR 10110000
bitwise NOT of a -10110011
bitwise NOT of b -11
bitwise LEFT 20220020
bitwise RIGHT 2527502
```

METHOD 2

DESCRIPTION:

In VSM, the corpus is represented in the form of the Term Document Matrix. Term Document Matrix represents documents vectors in matrix form in which the rows correspond to the terms in the document, columns correspond to the documents in the corpus and cells correspond to the weights of the terms.

CODE AND OUTPUT:

```
import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer

print("Boolean RetrievalModel USing Bitwise operations on Term Document Incidence Matrix")

corpus={'this is the first document' , 'this document is the second document', 'and this is the third document', 'is this the first document?'}

print("The corpus is: \n",corpus)

vectorizer=CountVectorizer()

x=vectorizer.fit_transform(corpus)

df=pd.DataFrame(x.toarray(),columns=vectorizer.get_feature_names_out()) #for newer version remove "_out"

print("The generated dataframe")

print(df)

print("Query processing on the term document incidence matrix")


#AND

print("Find all documents ids for query 'this' AND 'first'")

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

print("Document ids where with 'this' AND 'first' are present are: ",alldata.index.tolist())


#OR

print("Find all documents ids for query 'this' OR 'first'")

alldata=df[(df['this']==0)|(df['first']==1)]

print("Document ids where with 'this' OR 'first' are present are: ",alldata.index.tolist())
```

```
#NOT

print("Find all documents ids for query 'and' is not present")

alldata=df[(df['and']!=1)]

print("Document ids where with 'and' term are not present are: ",alldata.index.tolist())


#XOR

print("Find all documents ids for query 'this' XOR 'first'")

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

print("Document ids where with 'this' XOR 'first' are present are: ",alldata.index.tolist())
```

OUTPUT:

```
Boolean RetrievalModel USing Bitwise operations on Term Document Incidence Matrix
The corpus is:
 {'this is the first document', 'is this the first document?', 'this document is the second documen
t', 'and this is the third document'}
The generated dataframe
   and  document  first  is  second  the  third  this
0    0         1      1   1       0    1      0     1
1    0         1      1   1       0    1      0     1
2    0         2      0   1       1    1      0     1
3    1         1      0   1       0    1      1     1
Query processing on the term document incidence matrix
Find all documents ids for query 'this' AND 'first'
Document ids where with 'this' AND 'first' are present are:  [0, 1]
Find all documents ids for query 'this' OR 'first'
Document ids where with 'this' OR 'first' are present are:  [0, 1]
Find all documents ids for query 'and' is not present
Document ids where with 'and' term are not present are:  [0, 1, 2]
Find all documents ids for query 'this' XOR 'first'
Document ids where with 'this' XOR 'first' are present are:  [2, 3]
```

Information Retrieval
Practical No.2

DEPARTMENT OF COMPUTE SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	PageRank	**Batch**	I
Date:	3-1-24	Practical No	2

A) AIM: Implement Page Rank Algorithm.

METHOD I

B) DESCRIPTION:

About NetworkX : NetworkX is a Python package for the creation, manipulation of the structure ,dynamics and functions 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, it is no longer recommended. Installation The PyLab Module is installed at + as the Matplotlib package.

By the networkx package in python we can calculate page rank like below:

C) CODE AND OUTPUT:

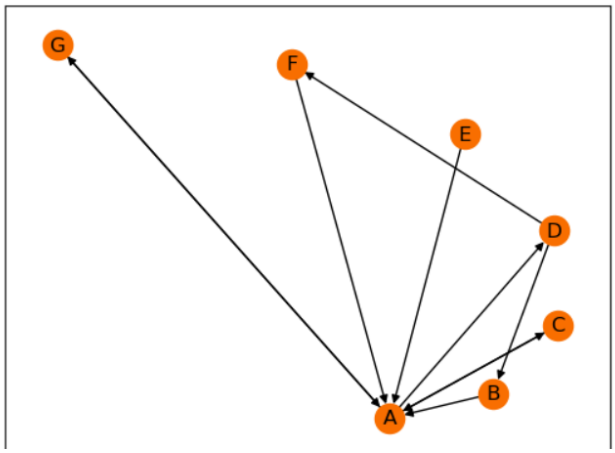
METHOD 1:

```
import networkx as nx

import pylab as plt

G=nx.DiGraph()

[G.add_node(k) for k in ["A","B","C","D","E","F","G"]]

G.add_edges_from([("G","A"),("A","G"),("B","A"),("A","D"),("D","B"),("A","C"),("C","A"),("D","F"),("F","A"),("E","A")])
```

```
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()
```

Output:

```
Page rank value {'A': 0.4080745143467559, 'B': 0.07967426232810562, 'C': 0.13704946318948705, 'D': 0.13704946318948705, 'E': 0.
021428571428571432, 'F': 0.07967426232810562, 'G': 0.13704946318948705}
```



METHOD 2

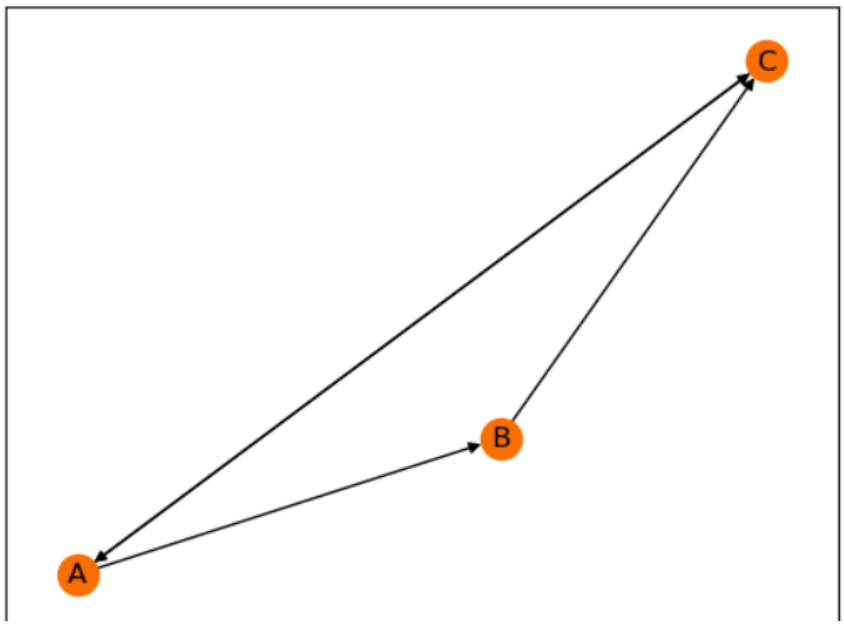
Implementation of PageRank using NetworkX

CODE AND OUTPUT:

```
import networkx as nx
import pylab as plt
G=nx.DiGraph()
[G.add_node(k) for k in ["A","B","C"]]
G.add_weighted_edges_from([('A','B',1),('A','C',1),('C','A',1),('B','C',1)])
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()
```

OUTPUT:

```
Page rank value {'A': 0.387789442707259, 'B': 0.21481051315058508, 'C': 0.3974000441421556}
```



METHOD 3

Code and output:

```
def page_rank(graph,damping_factor = 0.85, max_iterations = 100 ,tolerence = 1e-6):
    num_pages = len(graph)
    initial_page_rank = 1.0/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
        #Check convergence
        convergence= all(abs(new_page_ranks[page]-page_ranks[page])<tolerence for page in
graph)
        #update page ranks
        page_ranks=new_page_ranks
        if convergence:
            break
        return page_ranks
if __name__=="__main__":
        example_graph={
            'A':['B','C'],
            '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}")
```

Output:

```
Page: A -PageRank: 0.356250
```

Information Retrieval
Practical No.3

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Levenshtein Distance	**Batch**	I
Date:	09-01-2024	Practical No	3

A) AIM: Implement Dynamic programming algorithm for computing the edit distance between strings s1 and s2. (Hint. Levenshtein Distance)

B) DESCRIPTION:

The Levenshtein Distance measures the difference between two string sequences. It is the minimum number of edits needed to change or transform one string into the other. It is named after mathematician Vladimir Levenshtein who did a lot of research in field in the 1960s.

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 COMPUTE SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
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

import 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(doc2.similarity(doc3))
print(doc1.similarity(doc3))
```

Output:

```
1.0
0.582288958468651
0.582288958468651
```

METHOD 2: JACCARD SIMILARITY

```
def jaccard_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)
doc_1="Data is the new oil of the digital economy"
doc_2="Data is a new oil"
jaccard_Similarity(doc_1,doc_2)
```

Output:

```
0.444444444444444
```

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']
```

Information Retrieval
Practical No.5

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	Map Reduce Program	**Batch**	I
Date :	24-01-24	Practical No	5

A) AIM: Write a Map Reduce Program to count the number of occurrences of each alphabetic character in a given dataset.

B) DESCRIPTION:

MapReduce-

MapReduce is a big data analysis model that processes data sets using a parallel algorithm on computer clusters, typically Apache Hadoop clusters or cloud systems like Amazon Elastic MapReduce (EMR) clusters.

1. The map stage

The task of the map or mapper is to process the input data at this level. In most cases, the input data is stored in the Hadoop file system as a file or directory (HDFS). The mapper function receives the input file line by line. The mapper processes the data and produces several little data chunks.

2. The reduce stage (including shuffle and reduce)

The shuffle and reduce stages are combined to create the reduce stage. Processing the data that arrives from the mapper is the reducer's responsibility. The framework controls every aspect of data-passing, including assigning tasks, confirming their completion, and transferring data across nodes within a cluster. Most computing is done on nodes with data stored locally on drives, which lowers network traffic.

DEFAULTDICT -Dictionary in Python is an unordered collection of data values that are used to store data values like a map. Unlike other Data Types that hold only single value as an element, the Dictionary holds key-value pair.

REDUCE- In Python, reduce() is a built-in function that applies a given function to the elements of an iterable, reducing them to a single value.

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 MapReduce example."
    chunks=[chunk for chunk in dataset.split()]
    mapped_result=map(mapper,chunks)
    final_counts=reduce(reducer,mapped_result)
    for char, count in final_counts:
        print(f"Character: {char}, Count:{count}")
```

OUTPUT-

```
Character: h, Count:2
Character: e, Count:5
Character: l, Count:4
Character: o, Count:2
Character: w, Count:1
Character: r, Count:2
Character: d, Count:2
Character: t, Count:1
Character: i, Count:2
Character: s, Count:2
Character: a, Count:3
Character: m, Count:2
Character: p, Count:2
Character: u, Count:1
Character: c, Count:1
Character: x, Count:1
```

Information Retrieval
Practical No.6

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	HITS Algorithm	**Batch**	I
Date :	20-01-24	Practical No	6

A) AIM: HITS Algorithm.

B) DESCRIPTION:

Hyperlink Induced Topic Search (HITS) Algorithm is a Link Analysis Algorithm that rates webpages, developed by Jon Kleinberg. This algorithm is used to the web link-structures to discover and rank the webpages relevant for a particular search.

HITS uses hubs and authorities to define a recursive relationship between webpages.

Given a query to a Search Engine, the set of highly relevant web pages are called Roots. They are potential Authorities.

Pages that are not very relevant but point to pages in the Root are called Hubs. Thus, an Authority is a page that many hubs link to whereas a Hub is a page that links to many authorities

C) CODE AND OUTPUT:

```
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: 0.17909088824420202, 2: 0.410454555877899, 3: 0.410454555877899, 4: 7.153376910150593e-17, 5: -0.0}

Hub Scores: {1: -0.0, 2: 0.15188895956179282, 3: 0.15188895956179285, 4: 0.6962220808764141, 5: 1.2133715867828932e-16}

Information Retrieval
Practical No.7

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	Stopword Removal	**Batch**	I
Date :	24-01-24	Practical No	7

A) AIM: Write a program for Pre-processing of a Text Document: stop word removal.

B) DESCRIPTION:

Stopwords-

In **natural language processing (NLP)**, **stopwords** are frequently filtered out to enhance text analysis and computational efficiency. Eliminating stopwords can improve the accuracy and relevance of NLP tasks by drawing attention to the more important words, or content words. The article aims to explore stopwords

Punkt-

In NLTK, PUNKT is an unsupervised trainable model, which means it can be trained on unlabeled data. It generates a list of sentences from a text by developing a model for words that start sentences, prepositional phrases, and abbreviations using an unsupervised technique. Without first being put to use, it has to be trained on a sizable amount of plaintext in the intended language.

C) CODE AND OUTPUT:

Step1)

```
import nltk

nltk.download('stopwords')

from nltk.corpus import stopwords

set(stopwords.words('english'))
```

OUTPUT-

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

Step2) To tokenize and filter our sentence.

```
nltk.download('punkt')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
example="This is a sample sentence, showing off the stopwords filtration."
stop_words=set(stopwords.words('english'))
word_tokens=word_tokenize(example)
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', 'a', 'sample', 'sentence', ',', 'showing',
'off', 'the', 'stopwords', 'filtration', '.']
['This', 'sample', 'sentence', ',', 'showing', 'stopwords',
'filtration', '.']
```

SIES

Information Retrieval
Practical No.8

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	Mining Twitter	**Batch**	I
Date :	24-01-24	Practical No	8

A) AIM: Write a program for mining twitter to identify tweets for a specific period and identify trends and named entities.

B) DESCRIPTION:

Nitter Scraper:

Nitter Scraper is for anyone who enjoys the twitter-scraper library. Nitter Scraper leverages running a local docker container instance of nitter to scrape a users tweets and profile information without the twitter api ratelimit. This api works similar to the twitter-scraper project with a few differences.

Ntscraper is a package which is used to scrape twitter without any API.

C) CODE AND OUTPUT:

```
!pip install ntscraper


import pandas as pd

from ntscraper import Nitter

scraper=Nitter()

tweets=scraper.get_tweets('narendramodi',mode='user',number=5)
```

OUTPUT:

```
Testing instances: 100%|██████████████████████████████████████████████████████████| 31/31 [00:51<00:00,  1.66s/it]
24-Jan-24 10:30:14 - No instance specified, using random instance https://nitter.catsarch.com

24-Jan-24 10:30:20 - Current stats for narendramodi: 5 tweets, 0 threads...
```

```
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)
```

OUTPUT:

```
[['https://twitter.com/narendramodi/status/1749995168042987807#m', 'देशभर के मेरे परिवारजनों की ओर से जननायक कर्पूरी ठाकुर जी को उनकी जन्म-शताब्दी पर मेरी आदरपूर्ण श्रद्धांजलि। इस विशेष अवसर पर हमारी सरकार को उन्हें भारत रत्न से सम्मानित करने का सौभाग्य प्राप्त हुआ है। भारतीय समाज और राजनीति पर उन्होंने जो अविस्मरणीय छाप छोड़ी है, उसे लेकर मैं अपनी भावनाओं और विचारों को आपके साथ साझा कर रहा हूं… https://nm-4.com/vLEoBk', 'Jan 24, 2024 · 3:18 AM UTC', 9180, 449], ['https://twitter.com/narendramodi/status/1749994802488430667#m', 'I bow to Jan Nayak Karpoori Thakur Ji on his birth centenary. On this special occasion, our Government has had the honour of conferring the Bharat Ratna on him. I've penned a few thoughts on his unparalleled impact on our society and polity. https://nm-4.com/P8KL4m', 'Jan 24, 2024 · 3:17 AM UTC', 4134, 221], ['https://twitter.com/narendramodi/status/1749994107509112935#m', 'On National Girl Child Day, we salute the indomitable spirit and accomplishments of the Girl Child. We also recognise the rich potential of every girl child in all sectors. They are change-makers who make our nation and society better. Over the last decade, our government has been making many efforts to build a nation where every girl child has the opportunity to learn, grow and thrive.', 'Jan 24, 2024 · 3:14 AM UTC', 6722, 267], ['https://twitter.com/narendramodi/status/1749993137857245481#m', 'अध्यात्म, ज्ञान और शिक्षा की तपोभूमि उत्तर प्रदेश के अपने सभी परिवारजनों को राज्य के स्थापना दिवस की अनेकानेक शुभकामनाएं। बीते सात वर्षों में प्रदेश ने प्रगति की एक नई गाथा लिखी है, जिसमें राज्य सरकार के साथ जनता-जनार्दन ने भी बढ़-चढ़कर भागीदारी की है। मुझे विश्वास है कि विकसित भारत की संकल्प यात्रा में उत्तर प्रदेश अग्रणी भूमिका निभाएगा।', 'Jan 24, 2024 · 3:10 AM UTC', 6028, 357], ['https://twitter.com/narendramodi/status/1749810240030445643#m', 'मुझे इस बात की बहुत प्रसन्नता हो रही है कि भारत सरकार ने सामाजिक न्याय के पुरोधा महान जननायक कर्पूरी ठाकुर जी को भारत रत्न से सम्मानित करने का निर्णय लिया है। उनकी जन्म-शताब्दी के अवसर पर यह निर्णय देशवासियों को गौरवान्वित करने वाला है। पिछड़ों और वंचितों के उत्थान के लिए कर्पूरी जी की अटूट प्रतिबद्धता और दूरदर्शी नेतृत्व ने भारत के सामाजिक-राजनीतिक परिदृश्य पर अमिट छाप छोड़ी है। यह भारत रत्न न केवल उनके अतुलनीय योगदान का विनम्र सम्मान है, बल्कि इससे समाज में समरसता को और बढ़ावा मिलेगा।', 'Jan 23, 2024 · 3:04 PM UTC', 47939, 3506]]
```

```
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	9180	449
1	Jan 24, 2024 · 3:17 AM UTC	4134	221
2	Jan 24, 2024 · 3:14 AM UTC	6722	267
3	Jan 24, 2024 · 3:10 AM UTC	6028	357
4	Jan 23, 2024 · 3:04 PM UTC	47939	3506

Information Retrieval
Practical No.9

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	Web Crawling	**Batch**	I
Date :	29-01-24	Practical No	9

A) AIM: Write a program to implement simple web crawling.

B) DESCRIPTION:

Basic Operation:

1.The crawler begins with one or more URLs that constitute a seed set.

2.It picks a URL from this seed set, and then fetches the web page at the URL.

3.The fetched page is then parsed, to extract both the text and the links from the page (each pf which points to another URL).

4.The extracted text is fed to a text indexer.

5.The extracted links(URLs) are then added to a URL frontier, which at all times consists of URLs whose corresponding pages have yet to be fetched by the crawler.

6.Initially, the URL frontiercontains the seed set, as pages are fetched, the corresponding URLs are deleted from the URL frontier. The entire process may be viewed as traversing the web graph.

C) CODE AND OUTPUT:

```
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**********")

print(image_links)

print("**********/image_links**********")

end=time.time()

print("Time Taken in seconds:",(end-start))
```

OUTPUT-

```
**********Href_links**********

['#primary', 'http://recurship.com/', 'http://recurship.com/',
'http://recurship.com/', 'http://recurship.com/about/', 'http:/
/recurship.com/playthinks/', 'http://recurship.com/build-a-mvp/
', 'http://recurship.com/careers/', 'http://recurship.com/conta
ct/', 'http://recurship.com/blog/category/uncategorized/', 'htt
p://recurship.com/blog/2018/07/08/2018-7-8-sastaticket-acquires
-recurship/', 'http://recurship.com/blog/author/mashhoodr/', 'h
ttp://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-acqu
ires-recurship/', 'http://recurship.com/blog/category/uncategor
ized/', 'http://recurship.com/blog/2018/06/03/2018-6-4-ngrx-sel
ectors-how-to-stop-worrying-about-your-store-structure/', 'http
://recurship.com/blog/author/mashhoodr/', 'http://recurship.com
/blog/author/mashhoodr/', 'http://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-ngr
x-selectors-how-to-stop-worrying-about-your-store-structure/',
'http://recurship.com/blog/category/uncategorized/', 'http://re
curship.com/blog/2018/06/03/2018-6-1-jjknwadn9ivw1gba3wxsspjlpe
9grk/', 'http://recurship.com/blog/author/mashhoodr/', 'http://
recurship.com/blog/author/mashhoodr/', 'http://recurship.com/bl
og/2018/06/03/2018-6-1-jjknwadn9ivw1gba3wxsspjlpe9grk/', 'http:
```

//recurship.com/blog/2018/06/03/2018-6-1-jjknwadn9ivw1gba3wxsspjlpe9grk/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authentication-tool-belt/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authentication-tool-belt/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authentication-tool-belt/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-real-time-stream-processing-with-reactive-extensions-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-real-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://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/05/31/2018-5-31-supercharging-the-angular-cli-with-nx/', 'http://recurship.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/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/05/31/2018-5-31-angular-as-a-strategy-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/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/05/12/keynote-five-years-of-angular/', 'http://recurship.com/blog/2018/05/12/keynote-five-years-of-angular/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/04/29/2018-4-29-understanding-advanced-dependancy-injection-in-angular/', 'http://recurship.com/blog/author/ma

```
shhoodr/', '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-advanced-dependancy-injection-in-angular/', 'http://recurship.com/page/2/']

***********/href_links***********

['http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://recurship.com/wp-content/themes/stag-blocks/images/menu.svg', 'http://recurship.com/wp-content/themes/stag-blocks/images/close-button.svg', 'http://recurship.com/wp-content/themes/stag-blocks/images/search.svg', '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-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-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-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-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-blocks/images/back.svg']

***********/image_links***********

Time Taken in seconds: 0.33189868927001953
```

Information Retrieval
Practical No.10

DEPARTMENT OF COMPUTER SCIENCE

Name:	Kamal	**Roll Number**	TCS2324087
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	XML Retrieval	**Batch**	I
Date:	07-02-24	Practical No	10

A) AIM: Write a python program to parse XML text, generate Web graph and compute topic specific page rank.

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.

The webgraph is a directed graph, whose vertices correspond to the pages of the WWW, and a directed edge connects page X to page Y if there exists a hyperlink on page X, referring to page Y.

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:

```
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__":
    xml_data = """
    <webgraph>
        <page>
            <id>1</id>
            <link>2</link>
            <link>3</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>"""

    xml_root = parse_xml(xml_data)
    web_graph = generate_web_graph(xml_root)
    topic_specific_pagerank = compute_topic_specific_pagerank(web_graph,
topic_nodes=['1','2'])

    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}")
```

OUTPUT:

```
Topic-Specific PageRank
Node: 1 - PageRank: 0.350877
Node: 2 - PageRank: 0.350877
Node: 3 - PageRank: 0.298246
```