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. **CHAUHAN PANKAJ YAMUNAPRASAD**
Roll No **TCS2324007** 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-VI]**.

Prof. In-Charge
Prof. Rajesh Ramesh Yadav

Examination Date:
Examiner's Signature & Date:

HOD's Signature & Date:
Dr. Manoj Singh

College Seal
And
Date

Index Page

Sr. No	Description	Page No	Date	Faculty Signature
1	Write a python program to demonstrate bitwise operator	3	21/12/2023	
2	Implementation of Page Rank using NetworkX	8	3/1/2024	
3	Write a program to implement Levenshtein Distance.	13	10/1/2024	
4	Write a program to implement Jaccard Similarity. Write a program to implement Cosine Similarity.	15,17	10/1/2024	
5	Write a Python Program to implement a Map reducer.	19	24/1/2024	
6	Write a python program to implement HITS Algorithm	22	7/2/2024	
7	Write a Python Program for pre-processing of a text document: stopwords and removal	23	24/1/2024	
8	Write a program for mining Twitter to identify tweets for a specific period and identify trends and named entities.	32	24/1/2024	
9	Write a python program to implement a simple web crawler	38	29/1/24	
10	Write a program to parse XML text, generate Web graph and compute topic specific page rank.	41	7/2/2024	

SIES

Information Retrieval

Practical No.1

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
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 operator

B) DESCRIPTION:

Bitwise And:- Result bit 1,if both operand bits are 1;otherwise results bit

BITWISE OR:- Result bit 1,if any of the operand bit is 1; otherwise results bit 0

BITWISE NOT:- Inverts individual bits

BITWISE LEFTSHIFT:- The left operand's value is moved toward left by the number of bits specified by the right operand.

BITWISE RIGHTSHIFT:- The left operand's value is moved toward right by the number bits specified by the right operand

Method 1

Code: -

[1]:
```
def bitwise_operation(a,b):
 #bitwise AND
 bitwise_and_result = a&b

 #bitwise OR
 bitwise_or_result = a|b

 #bitwise XOR
 bitwise_xor_result = a^b

 #bitwise NOT
 bitwise_not_result = a=-b

 #bitwise Left Shift
 bitwise_left_shift_result = a<<1,b<<1

 #bitwise Right Shift
 bitwise_right_shift_result = a>>1,b>>1
 print(f"Bitwise AND: {bitwise_and_result}")
 print(f"Bitwise OR: {bitwise_or_result}")
 print(f"Bitwise XOR: {bitwise_xor_result}")
 print(f"Bitwise NOT of a: {bitwise_not_result}")
 print(f"Bitwise NOT of b: {bitwise_not_result}")
 print(f"Bitwise Left Shift of a: {bitwise_left_shift_result}")
 print(f"Bitwise Left Shift of b: {bitwise_left_shift_result}")
 print(f"Bitwise Right Shift of a: {bitwise_right_shift_result}")
 print(f"Bitwise Right Shift of b: {bitwise_right_shift_result}")

a = int(input("enter the binary number: "))
b = int(input("enter the binary number: "))
print(bitwise_operation(a,b))
```

Output: -

```
enter the binary number:  1001
enter the binary number:  1100
Bitwise AND: 72
Bitwise OR: 2029
Bitwise XOR: 1957
Bitwise NOT of a: -1100
Bitwise NOT of b: -1100
Bitwise Left Shift of a: (-2200, 2200)
Bitwise Left Shift of b: (-2200, 2200)
Bitwise Right Shift of a: (-550, 550)
Bitwise Right Shift of b: (-550, 550)
None
```

Method 2

Code: -

```
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', 'an this is the third document', 'Is this is the First 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   1         1      0   1       0    1      1     1
3   0         1      1   2       0    1      0     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: [3]

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, 2]
```

SIES

Information Retrieval

Practical No.2

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TY B.Sc(Computer Science)
Topic:	Page Rank	**Batch**	I
Date:	3/1/2024	**Practical No**	2

A) AIM: Implementation of Page Rank using NetworkX.

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: (Without using Weighted Edges)

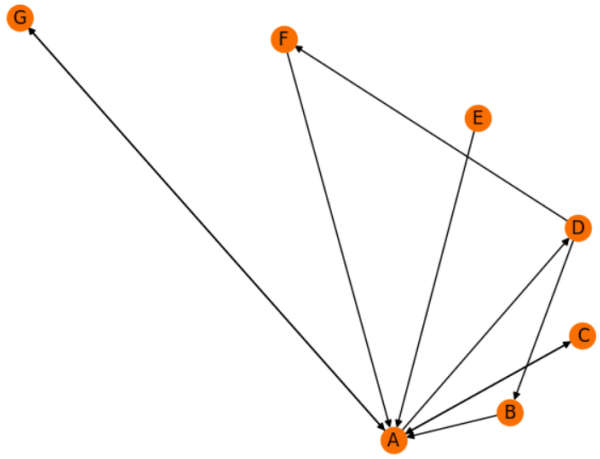
CODE:

[1]:
```
import networkx as nx
import matplotlib.pyplot 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'), ('C', 'A'), ('A', 'C'), ('A', 'D'), ('D', 'B'), ('D', 'F'), ('F', 'A'), ('E', 'A')])
ppr1 = nx.pagerank(G)
print("Page rank value:", ppr1)
pos = nx.spiral_layout(G)
nx.draw(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}
```



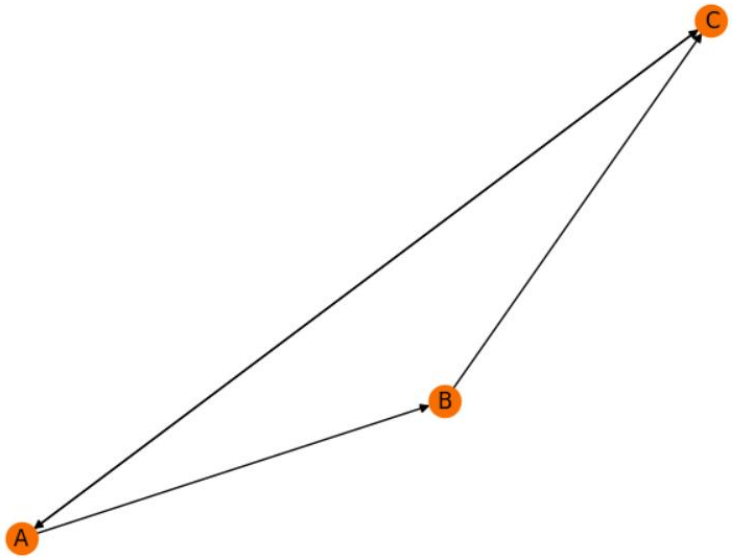
METHOD 2: (Using Weighted Edges)

CODE:

```
[1]: import networkx as nx
     import pylab as plt
     G=nx.DiGraph()
     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(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: (Using Solved Example)

CODE:

```
•[1]: #Method 3: page rank
def page_rank(graph,damping_factor=0.85,max_iterations=100,tolerance=1e-6):
    num_pages=len(graph)
    initial_page_rank=1.0/num_pages
    #intialize page ranks
    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 convegance-to stop the loop
            convergence=all(abs(new_page_ranks[page]-page_ranks[page])<tolerance for page in graph)
            #Update page ranks
            page_ranks=new_page_ranks
            if convergence:
                break
            return page_ranks
example_graph={
    'A':['B','C'],
    'B':['A'],
    'C':['A','B'],
    'D':['B']
}
#Calculate page rank
result= page_rank(example_graph)
print(result)
#Print PageRank results
for page,rank in sorted(result.items(),key=lambda x:x[1],reverse=True):
    print(f"Page:{page} - PageRank:{rank:4f}")
```

Output: -

```
{'A': 0.35625}
Page:A - PageRank:0.356250
```

SIES

Information Retrieval

Practical No.3

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TY B.Sc(Computer Science)
Topic:	Levenshtein Distance	**Batch**	I
Date :	10/1/2024	Practical No	3

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

B) DESCRIPTION:

Levenshtein Distance, also known as Edit Distance, is a measure of similarity between two strings in terms of the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one string into the other

C) CODE AND OUTPUT:

CODE:

```
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,  # insert
                A[i+1][j] + 1,  # replace
                A[i][j] + int(x[i] != y[j])  # delete
            )
    return A[n][m]

print(leven("brap", "rap"))
print(leven("trial", "try"))
print(leven("horse", "force"))
print(leven("abcd", "aecdb"))
print(leven("monkey", "money"))
```

Output: -

```
1
3
2
2
1
```

SIES

Information Retrieval

Practical No.4a

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TY B.Sc(Computer Science)
Topic:	Jaccard Similarity	**Batch**	I
Date :	10/1/2024	Practical No	4a

A) AIM: Write a program to implement Jaccard Similarity.

B) DESCRIPTION:

The Jaccard Similarity is a measure used to compare the similarity and dissimilarity between two sets. It is defined as the size of the intersection of the sets divided by the size of the union of the sets.

C) CODE AND OUTPUT:

METHOD 1

CODE:

```
[1]: #jaccard similarity
     def jaccard(d1, d2):
      wd1 = set(d1.lower().split())
      wd2 = set(d2.lower().split())
      intersection = wd1.intersection(wd2)
      union = wd1.union(wd2)
      return float(len(intersection))/len(union)
     d1 = "Data is the new oil of digital economy"
     d2 = "Data is a new oil"
     jaccard(d1, d2)
```

Output: -

```
[1]: 0.4444444444444444
```

SIES

Information Retrieval

Practical No.4b

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TY B.Sc(Computer Science)
Topic:	Cosine Similarity	**Batch**	I
Date :	10/1/2024	Practical No	4b

A) AIM: Write a program to implement Cosine Similarity.

B) DESCRIPTION:

Cosine Similarity is a measure used to determine how similar two non-zero vectors are irrespective of their sizes. In the context of natural language processing or information retrieval, it's commonly applied to compare the similarity of documents represented as vectors in a high-dimensional space, where each dimension corresponds to a term's frequency in the document.

C) CODE AND OUTPUT:

METHOD 1

CODE:

```
[1]: #Cosine Similarity
d1 = "Data is the new oil of digital economy"
d2 = "Data is a new oil"
data = [d1, d2]
from sklearn.metrics.pairwise import cosine_similarity
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_df=(vector_matrix.toarray(),tokens)
cosine_similarity_matrix = cosine_similarity(vector_matrix)
create_df = (cosine_similarity_matrix,['d1','d2'])
print(create_df)
```

Output: -

```
(array([[1.        , 0.57973867],
       [0.57973867, 1.        ]]), ['d1', 'd2'])
```

SIES

Information Retrieval

Practical No.5

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Reduce doc	**Batch**	I
Date:	24-1-24	Practical No	5

A) AIM: Write a Python Program to implement a Map reducer.

B) DESCRIPTION:

MapReduce is a processing technique and a program model for distributed computing based on java. The MapReduce algorithm contains two important tasks, namely Map and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs)

Code: -

[1]:
```python
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 map reduce example"
    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: 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
```

SIES

Information Retrieval

Practical No.6

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Reduce doc	**Batch**	I
Date:	24-1-24	Practical No	6

A) AIM: Write a python program to implement 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. Before understanding the HITS Algorithm, we first need to know about Hubs and Authorities.

- 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

Code: -

[1]:

```
#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: -1.1515955496845525, 2: 1.0757977748422762, 3: 1.0757977748422762, 4: -7.489742195924957e-17, 5: -0.0}
Hub Scores: {1: -3.9524865797234856e-15, 2: 7.596499713090878, 3: 7.5964997130908785, 4: -14.192999426181753, 5: 9.88121644930872e-16}
```

SIES

Information Retrieval

Practical No.7

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	STOPWORDS	**Batch**	I
Date:	24-1-24	Practical No	7a & 7b

A) AIM: Write a Python Program for pre-processing of a text document: stopwords and removal

B) DESCRIPTION:

Tokenization is the process by which a large quantity of text is divided into smaller parts called tokens. These tokens are very useful for finding patterns and are considered as a base step for stemming and lemmatization. Tokenization also helps to substitute sensitive data elements with non-sensitive data elements1

C) CODE AND OUTPUT:

Practical no 7a

Code: -

```
[1]: import nltk
     nltk.download('stopwords')
     from nltk.corpus import stopwords
     set(stopwords.words('english'))
```

Output: -

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\panka\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
[1]: {'a',
      'about',
      'above',
      'after',
      'again',
      'against',
```

{'a',
'about',
'above',
'after',
'again',
'against',
'ain',
'all',
'am',
'an',
'and',
'any',
'are',
'aren',
"aren't",
'as',
'at',
'be',
'because',
'been',
'before',
'being',
'below',
'between',
'both',
'but',
'by',
'can',
'couldn',
"couldn't",
'd',
'did',
'didn',
"didn't",
'do',
'does',
'doesn',

"doesn't",
'doing',
'don',
"don't",
'down',
'during',
'each',
'few',
'for',
'from',
'further',
'had',
'hadn',
"hadn't",
'has',
'hasn',
"hasn't",
'have',
'haven',
"haven't",
'having',
'he',
'her',
'here',
'hers',
'herself',
'him',
'himself',
'his',
'how',
'i',
'if',
'in',
'into',
'is',
'isn',
"isn't",
'it',
"it's",
'its',
'itself',
'just',
'll',
'm',
'ma',
'me',
'mightn',
"mightn't",
'more',
'most',
'mustn',
"mustn't",
'my',
'myself',
'needn',
"needn't",
'no',
'nor',
'not',

'now',
'o',
'of',
'off',
'on',
'once',
'only',
'or',
'other',
'our',
'ours',
'ourselves',
'out',
'over',
'own',
're',
's',
'same',
'shan',
"shan't",
'she',
"she's",
'should',
"should've",
'shouldn',
"shouldn't",
'so',
'some',
'such',
't',
'than',
'that',
"that'll",
'the',
'their',
'theirs',
'them',
'themselves',
'then',
'there',
'these',
'they',
'this',
'those',
'through',
'to',
'too',
'under',
'until',
'up',
've',
'very',
'was',
'wasn',
"wasn't",
'we',
'were',
'weren',
"weren't",

'what',
'when',
'where',
'which',
'while',
'who',
'whom',
'why',
'will',
'with',
'won',
"won't",
'wouldn',
"wouldn't",
'y',
'you',
"you'd",
"you'll",
"you're",
"you've",
'your',
'yours',
'yourself',
'yourselves'}

C) CODE AND OUTPUT:

Practical no 7b

Code: -

[28]:
```
import nltk
nltk.download('punkt')
nltk.download('stopwords')

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

example_sent = "this is a 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]

# Alternatively, you can use the following loop for filtering
# filtered_sentence = []
# for w in word_tokens:
#     if w not in stop_words:
#         filtered_sentence.append(w)

print(word_tokens)
print(filtered_sentence)
```

Output: -

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\panka\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
['this', 'is', 'a', 'sample', 'sentence', ',', 'showing', 'off', 'the', 'stop', 'words', 'filtration']
['sample', 'sentence', ',', 'showing', 'stop', 'words', 'filtration']
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\panka\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Information Retrieval

Practical No.8

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	Roll Number	TCS2324007
Paper Code:	SIUSCS64	Class	TYBSc(Computer Science)
Topic:	Mining twitter	Batch	I
Date:	24-1-24	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:

Automated data mining is the process of using computer algorithms to automatically extract patterns and insights from large datasets. Python is a popular programming language for automated data mining due to its ease of use, large range of libraries, and powerful data analysis capabilities.

C) CODE AND OUTPUT:

```
import pandas as pd

from ntscraper import Nitter

scraper=Nitter()
```

__Output:__

Testing instances: 100%|██| 31/31 [00:43<00:00, 1.42s/it]

C) CODE AND OUTPUT:

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

tweets

Output:

24-Jan-24 10:31:33- No instance specified, using random instance https://nitter.perennialte.ch

24-Jan-24 10:31:40- Current stats for narendramodi: 5 tweets, 0 threads...

{'tweets': [{'link': 'https://twitter.com/narendramodi/status/1749995168042987807#m',

'text': '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',

'user': {'name': 'Narendra Modi',

'username': '@narendramodi',

'profile_id': '1700051019525488640',

'avatar': 'https://pbs.twimg.com/profile_images/1700051019525488640/VRqy0bTE_bigger.jpg'},

'date': 'Jan 24, 2024 · 3:18 AM UTC',

'is-retweet': False,

'external-link': '',

'replying_to': [],

'quoted-post': {},

'stats': {'comments': 451, 'retweets': 1803, 'quotes': 43, 'likes': 9266},

'pictures': [],

'videos': [],

'gifs': []},

{'link': 'https://twitter.com/narendramodi/status/1749994802488430667#m',

'text': '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',

'user': {'name': 'Narendra Modi',

'username': '@narendramodi',

'profile_id': '1700051019525488640',

'avatar': 'https://pbs.twimg.com/profile_images/1700051019525488640/VRqy0bTE_bigger.jpg'},

'date': 'Jan 24, 2024 · 3:17 AM UTC',

'is-retweet': False,

'external-link': '',

'replying_to': [],

'quoted-post': {},

'stats': {'comments': 221, 'retweets': 1062, 'quotes': 28, 'likes': 4162},

'pictures': [],

'videos': [],

'gifs': []},

{'link': 'https://twitter.com/narendramodi/status/1749994107509112935#m',

'text': '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.',

'user': {'name': 'Narendra Modi',

'username': '@narendramodi',

'profile_id': '1700051019525488640',

'avatar': 'https://pbs.twimg.com/profile_images/1700051019525488640/VRqy0bTE_bigger.jpg'},

'date': 'Jan 24, 2024 · 3:14 AM UTC',

'is-retweet': False,

'external-link': '',

'replying_to': [],

'quoted-post': {},

'stats': {'comments': 270, 'retweets': 1247, 'quotes': 32, 'likes': 6790},

'pictures': [],

'videos': [],

'gifs': []},

{'link': 'https://twitter.com/narendramodi/status/1749993137857245481#m',

'text': '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।',

'user': {'name': 'Narendra Modi',

'username': '@narendramodi',

'profile_id': '1700051019525488640',

'avatar': 'https://pbs.twimg.com/profile_images/1700051019525488640/VRqy0bTE_bigger.jpg'},

'date': 'Jan 24, 2024 · 3:10 AM UTC',

'is-retweet': False,

'external-link': '',

'replying_to': [],

'quoted-post': {},

'stats': {'comments': 357, 'retweets': 1433, 'quotes': 34, 'likes': 6062},

'pictures': [],

'videos': [],

'gifs': []},

{'link': 'https://twitter.com/narendramodi/status/1749810240030445643#m',

'text': 'मुझे इस बात की बहुत प्रसन्नता हो रही है कि भारत सरकार ने समाजिक 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।',

'user': {'name': 'Narendra Modi',

'username': '@narendramodi',

'profile_id': '1700051019525488640',

'avatar': 'https://pbs.twimg.com/profile_images/1700051019525488640/VRqy0bTE_bigger.jpg'},

'date': 'Jan 23, 2024 · 3:04 PM UTC',

'is-retweet': False,

'external-link': '',

'replying_to': [],

'quoted-post': {},

'stats': {'comments': 3506,

'retweets': 8668,

```
  'quotes': 428,
  'likes': 47985},
 'pictures': ['https://pbs.twimg.com/media/GEiTGDebMAAKNBB.jpg'],
 'videos': [],
 'gifs': []}],
'threads': []}
```

C) CODE AND OUTPUT:

```
final_tweets=[]
for tweet in tweets['tweets']:
    data=[tweet['link'],tweet['text'],tweet['date'],tweet['stats']['likes']]
    final_tweets.append(data)
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',
  9266],
 ['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',
  4162],
 ['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',
```

C) CODE AND OUTPUT:

```
data=pd.DataFrame(final_tweets,columns=['link','text','date','No'])
data
```

Output:

	link	text	date	No
0	https://twitter.com/narendramodi/status/174999...	देशभर के मेरे परिवारजनों की ओर से जननायक कर्पू...	Jan 24, 2024 · 3:18 AM UTC	9266
1	https://twitter.com/narendramodi/status/174999...	I bow to Jan Nayak Karpoori Thakur Ji on his b...	Jan 24, 2024 · 3:17 AM UTC	4162
2	https://twitter.com/narendramodi/status/174999...	On National Girl Child Day, we salute the indo...	Jan 24, 2024 · 3:14 AM UTC	6790
3	https://twitter.com/narendramodi/status/174999...	अध्यात्म, ज्ञान और शिक्षा की तपोभूमि उत्तर प्र...	Jan 24, 2024 · 3:10 AM UTC	6062
4	https://twitter.com/narendramodi/status/174981...	मुझे इस बात की बहुत प्रसन्नता हो रही है कि भार...	Jan 23, 2024 · 3:04 PM UTC	47985

SIES

Information Retrieval

Practical No.9

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	TYBSc(Computer Science)
Topic:	Web Crawler	**Batch**	I
Date:	29-1-24	Practical No	9

A) AIM: Write a python program to implement a simple web crawler

B) DESCRIPTION: A web crawler is a digital search engine bot that uses copy and metadata to discover and index site pages. Also referred to as a spider bot, it "crawls" the world wide web to learn what a given page is about. It then indexes the pages and stores the information for future searches.

Working of web crawler:

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 pages at that URL.
3. The fetched page is then parsed, to extract both the text and the links from the page.
4. The extracted text is fed to a text indexer.
5. The extracted links and 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 frontier contains 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,.

Code: -

```
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 link ****************************")
print(href_links)
print("******************/href_links****************************")
print("****************** Image Link ****************************")
print(image_links)
print("******************/image_links****************************")

end = time.time()
print("Time taken in seconds: ", (end - start))
```

Output: -

```
8/06/03/2018-6-1-jjknwadn9ivw1gba3wxsspjlpe9grk/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://
recurship.com/blog/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://re
curship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authent
ication-tool-belt/', 'http://recurship.com/blog/2018/06/03/2018-5-31-angulars-user-authentication-tool-belt/', 'http://recurship.com/blog/category/uncate
gorized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recur
ship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/06/03/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.com/blog/2018/06/0
3/2018-5-31-xfvrq9aauqkayhkd4kzp7gsbfg2bfl/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/06/03/2018-5-31-real-t
ime-stream-processing-with-reactive-extensions-rx/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'htt
p://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://recurship.com/blog/author/mashhoodr/', 'http://recur
ship.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/2018/05/31/2018-5-3
1-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-scal
e/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/05/12/keynote-five-years-of-angular/', 'http://recurship.com/bl
og/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/2018/05/12/keynote-five-years-of-angular/', 'http://recu
rship.com/blog/2018/05/12/keynote-five-years-of-angular/', 'http://recurship.com/blog/category/uncategorized/', 'http://recurship.com/blog/2018/04/29/201
8-4-29-understanding-advanced-dependancy-injection-in-angular/', 'http://recurship.com/blog/author/mashhoodr/', 'http://recurship.com/blog/author/mashhoo
dr/', 'http://recurship.com/blog/2018/04/29/2018-4-29-understanding-advanced-dependancy-injection-in-angular/', 'http://recurship.com/blog/2018/04/29/201
8-4-29-understanding-advanced-dependancy-injection-in-angular/', 'http://recurship.com/page/2/']
******************/href_links*****************************
****************** Image Link *****************************
['http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://recurship.com/wp-content/themes/stag-blocks/images/menu.svg', 'htt
p://recurship.com/wp-content/themes/stag-blocks/images/close-button.svg', 'http://recurship.com/wp-content/themes/stag-blocks/images/search.svg', 'htt
p://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=8
0&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/8a081ac7e6aadaabfdc51ec0388678
90?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag-blocks/images/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51ec0388
67890?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/placeholder.svg', 'http://2.gravatar.com/avatar/8a081ac7e6aadaabfdc51
ec038867890?s=80&d=mm&r=g', 'http://recurship.com/wp-content/themes/stag-blocks/images/back.svg']
******************/image_links*****************************
Time taken in seconds:  0.24193835258483887
```

Information Retrieval

Practical No.10

DEPARTMENT OF COMPUTER SCIENCE

Name:	CHAUHAN PANKAJ YAMUNAPRASAD	**Roll Number**	TCS2324007
Paper Code:	SIUSCS64	**Class**	B.Sc(Computer Science)
Topic:	Information Retrieval	**Batch**	I
Date :	7/2/24	Practical No	10

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

B) DESCRIPTION:

XML retrieval breaks away from the traditional retrieval unit of a document as a single large (text) block and aims to implement focused retrieval strategies aiming at returning document components, i.e., XML elements, instead of whole documents in response to a user query.

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__":
    # Example XML text representing web pages and links
    example_xml = '''
    <webpage>
    <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>
    <page>
    <id>3</id>
    <link>1</link>
    <link>2</link>
    </page>
    </webpage>
    '''
    # Parse XML
    xml_root = parse_xml(example_xml)
    # Generate web graph
    web_graph = generate_web_graph(xml_root)
    topic_specific_pagerank = compute_topic_specific_pagerank(web_graph, topic_nodes=['1', '2'])
    # Print the results
    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 : 0.3509
Node : 2 - PageRank : 0.3509
Node : 3 - PageRank : 0.2982
```