

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

**CERTIFICATE**

This is to certify that Mr./ Mrs. **R Vishwa Varshini** Roll No. **FMSC2425181** has successfully completed the necessary course of experiments in the subject of **Web Mining** during the academic year **2024 – 2025** complying with the requirements for the course of **M.Sc. Information Technology Part 1 [Semester-II]**

Prof. In-Charge Head of Department

**Prof. Rajesh Yadhav Prof. Sudha B**

Examination Date:

Examiner’s Signature & Date:

College Seal & Date

INDEX

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Practical** | **Pg. No.** | **Sign** |
| **01** | **Write a python program to implement Link Analysis.** | **03** |  |
| **02** | **Write a python program to make a Web Graph.** | **05** |  |
| **03** | **Write a python program to perform simple crawling from Wikipedia using keyword.** | **07** |  |
| **04** | **Write a python program to fetch page content.** | **09** |  |
| **05** | **Write a python program for HITS algorithm.** | **11** |  |
| **06** | **Write a python program for Page Rank.** | **13** |  |
| **07** | **Write a python program to summarize text.** | **15** |  |
| **08** | **Write a python program to implement recommender system** | **18** |  |

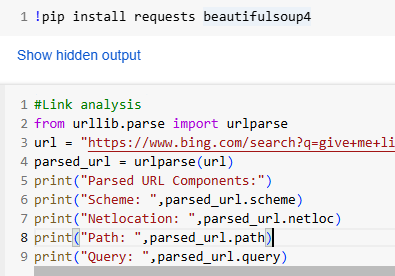


(Web Mining - II) Practical No. 1

# DEPARTMENT OF INFORMATION TECHNOLOGY

|  |
| --- |
| 1. **AIM: Write a python program for Link analysis** 2. **DESCRIPTION:**   Link analysis is a data analysis technique that examines the relationships between data points in a network. It's also known as network analysis.   1. **CODE AND OUTPUT:**   **!pip install requests beautifulsoup4 #Link analysis**  **from urllib.parse import urlparse url =**  **"https://**[**www.bing.com/search?q=give+me+links+to+check+in+link+analysis&qs=n&form=QBRE**](http://www.bing.com/search?q=give%2Bme%2Blinks%2Bto%2Bcheck%2Bin%2Blink%2Banalysis&qs=n&form=QBRE) **&sp=-1&lq=0&pq=give+me+links+to+check+in+link+analysis&sc=0- 39&sk=&cvid=B96605B98AAD4BD2B59BAEF6EB7C33A3&ghsh=0&ghacc=0&ghpl="**  **parsed\_url = urlparse(url) print("Parsed URL Components:") print("Scheme: ",parsed\_url.scheme) print("Netlocation: ",parsed\_url.netloc) print("Path: ",parsed\_url.path) print("Query: ",parsed\_url.query)** |

## Output:



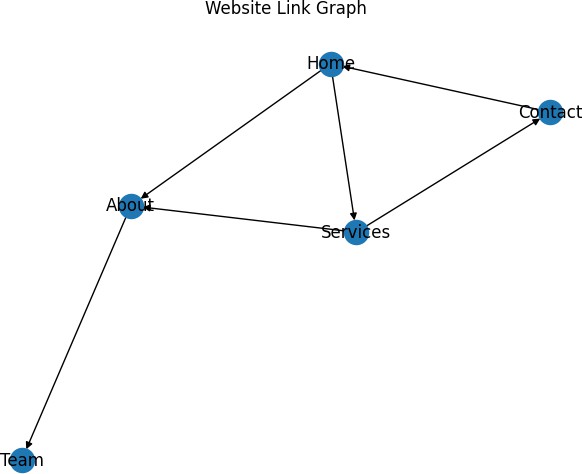
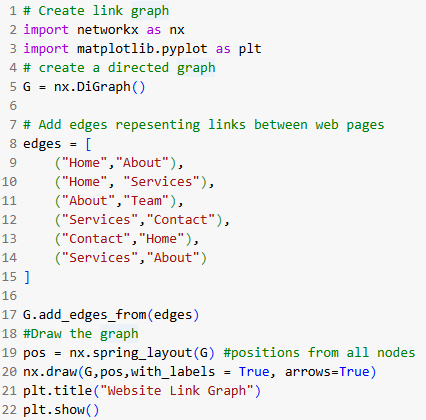


(Web Mining - II) Practical No. 2

# DEPARTMENT OF INFORMATION TECHNOLOGY

|  |
| --- |
| 1. **AIM: Write a python program to make a web graph** 2. **DESCRIPTION:**   A web graph is a visual representation of the World Wide Web (WWW) that shows how web pages are linked together. It's a directed graph, which means that the edges are directed lines.   1. **CODE AND OUTPUT:**   **# Create link graph import networkx as nx**  **import matplotlib.pyplot as plt # create a directed graph**  **G = nx.DiGraph()**  **# Add edges representing links between web pages edges = [**  **("Home","About"),**  **("Home", "Services"),**  **("About","Team"),**  **("Services","Contact"),**  **("Contact","Home"),**  **("Services","About")**  **]**  **G.add\_edges\_from(edges) #Draw the graph** |

pos = nx.spring\_layout(G) #positions from all nodes nx.draw(G,pos,with\_labels = True, arrows=True) plt.title("Website Link Graph")



plt.show()

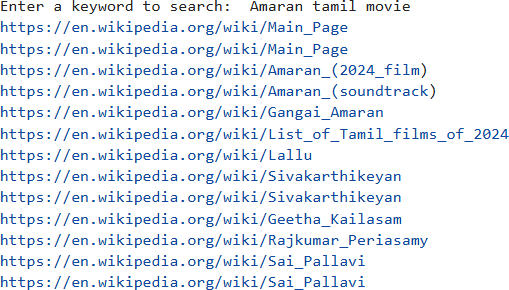
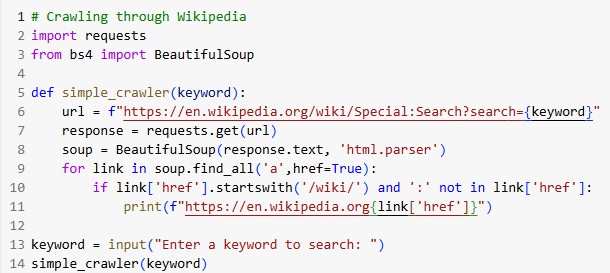
## Output:



(Web Mining - II) Practical No. 3

# DEPARTMENT OF INFORMATION TECHNOLOGY

|  |
| --- |
| 1. **AIM: Write a python program to perform simple crawling from Wikipedia using keywords** 2. **DESCRIPTION:**   A web crawler is a computer program that automatically browses the web to index web pages and their content. Web crawlers are also known as bots, spiders, or web robots.   1. **CODE AND OUTPUT:**   **# Crawling through Wikipedia import requests**  **from bs4 import BeautifulSoup**  **def simple\_crawler(keyword):**  **url = f"https://en.wikipedia.org/wiki/Special:Search?search={keyword}" response = requests.get(url)**  **soup = BeautifulSoup(response.text, 'html.parser') for link in soup.find\_all('a',href=True):**  **if link['href'].startswith('/wiki/') and ':' not in link['href']: print(f"https://en.wikipedia.org{link['href']}")**  **keyword = input("Enter a keyword to search: ") simple\_crawler(keyword)** |



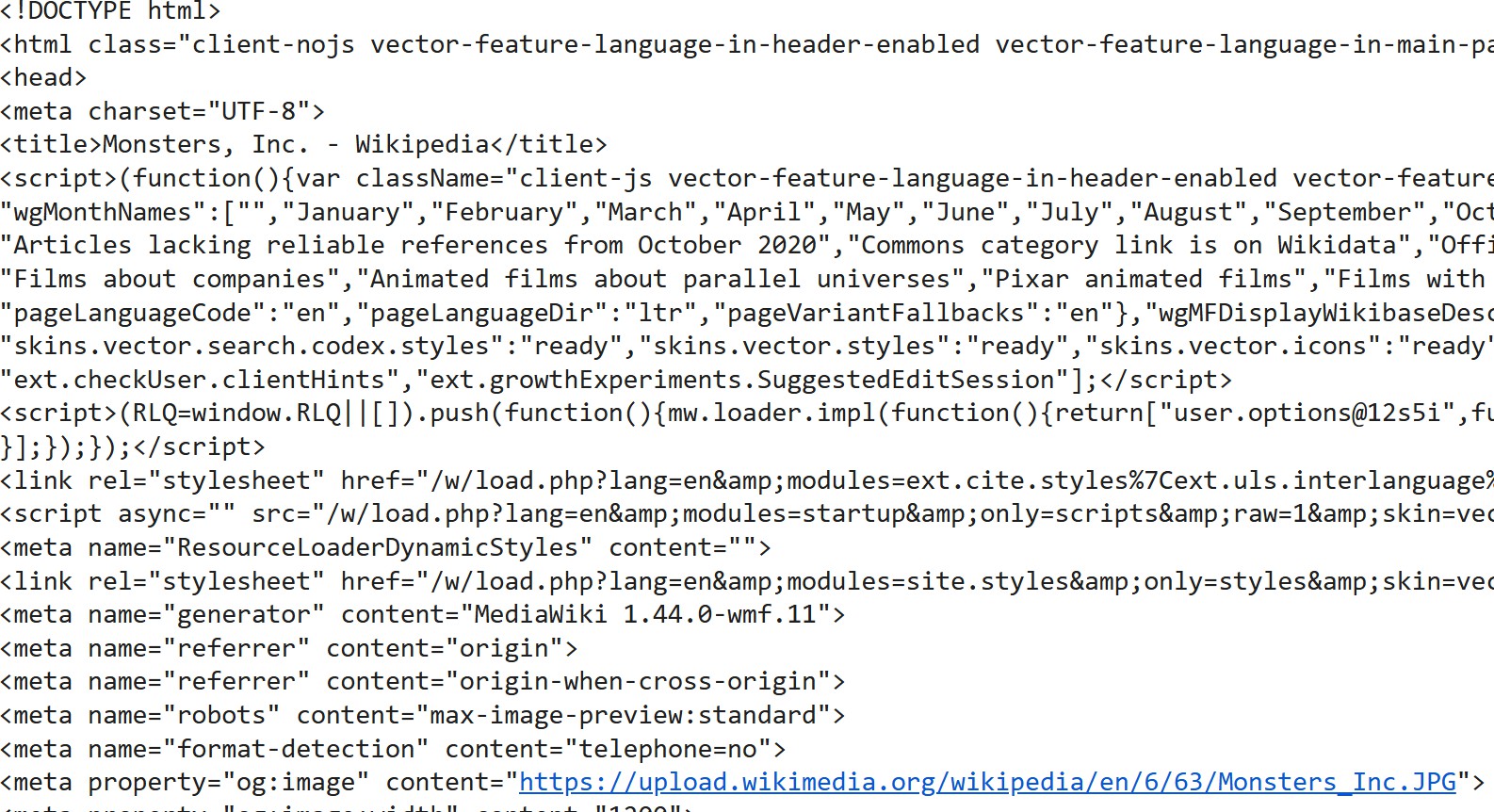
## Output:



(Web Mining - II) Practical No. 4

# DEPARTMENT OF INFORMATION TECHNOLOGY

|  |
| --- |
| 1. **AIM: Write a python program to fetch page content** 2. **DESCRIPTION:**   Fetching a page is the process of retrieving data from a source, such as a database or the internet. It's an important concept in computing that allows websites to retrieve only the data they need.   1. **CODE AND OUTPUT:**   **# Fetch page import requests**  **def fetch\_page(url):**  **headers = {'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64)'} try:**  **response = requests.get(url, headers=headers) response.raise\_for\_status() # Raise an HTTPError for bad responses return response.text**  **except requests.exceptions.HTTPError as e: print(f"HTTP error occurred: {e}")**  **except Exception as e:**  **print(f"An error occurred: {e}") url="https://en.wikipedia.org/wiki/Monsters,\_Inc." # Replace with the target URL page\_content = fetch\_page(url)**  **if page\_content:**  **print(page\_content)** |



## Output:



(Web Mining - II) Practical No. 5

# DEPARTMENT OF INFORMATION TECHNOLOGY

|  |
| --- |
| 1. **AIM: Write a python program for Hits Algorithm** 2. **DESCRIPTION:**   The Hyperlink-Induced Topic Search (HITS) algorithm is a link analysis algorithm that ranks web pages based on their relevance to a topic. It's also known as "hubs and authorities".   1. **CODE AND OUTPUT:**   **# HITS**  **import networkx as nx G = nx.DiGraph()**  **# G.add\_edges\_from([(1,2), (1,3), (2,4), (3,4), (4,5)])**  **G.add\_edges\_from([('p','q'), ('p','s'), ('p','r'), ('q','r'), ('r','p'), ('s','s')]) authority\_score, hub\_score = nx.hits(G)**  **print(f"Authority Scores: {authority\_score}")**  **print(f"Hub Scores: {hub\_score}")** |

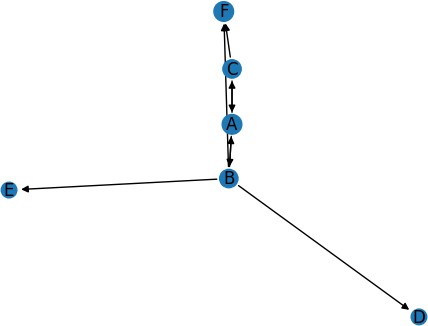
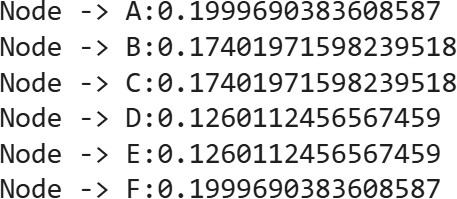
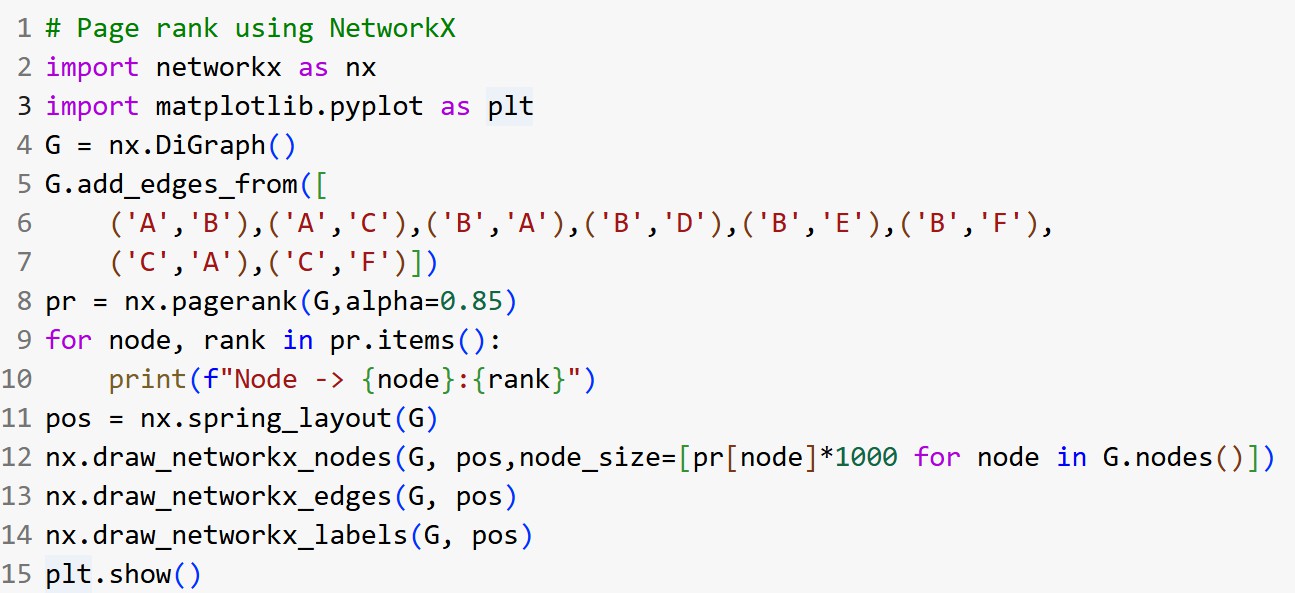
|  |
| --- |
|  |
| **Output:** |
|  |



(Web Mining - II) Practical No. 6

# DEPARTMENT OF INFORMATION TECHNOLOGY

|  |
| --- |
| 1. **AIM: Write a python program for Page Rank** 2. **DESCRIPTION:**   **PageRank (PR) is an algorithm that Google uses to rank web pages in search results. It measures the importance of a page by counting the number and quality of links that point to it.**   1. **CODE AND OUTPUT:**   **# Page rank using NetworkX import networkx as nx**  **import matplotlib.pyplot as plt G = nx.DiGraph() G.add\_edges\_from([**  **('A','B'),('A','C'),('B','A'),('B','D'),('B','E'),('B','F'), ('C','A'),('C','F')])**  **pr = nx.pagerank(G,alpha=0.85) for node, rank in pr.items():**  **print(f"Node -> {node}:{rank}") pos = nx.spring\_layout(G)**  **nx.draw\_networkx\_nodes(G, pos,node\_size=[pr[node]\*1000 for node in G.nodes()]) nx.draw\_networkx\_edges(G, pos)**  **nx.draw\_networkx\_labels(G, pos)**  **plt.show()** |



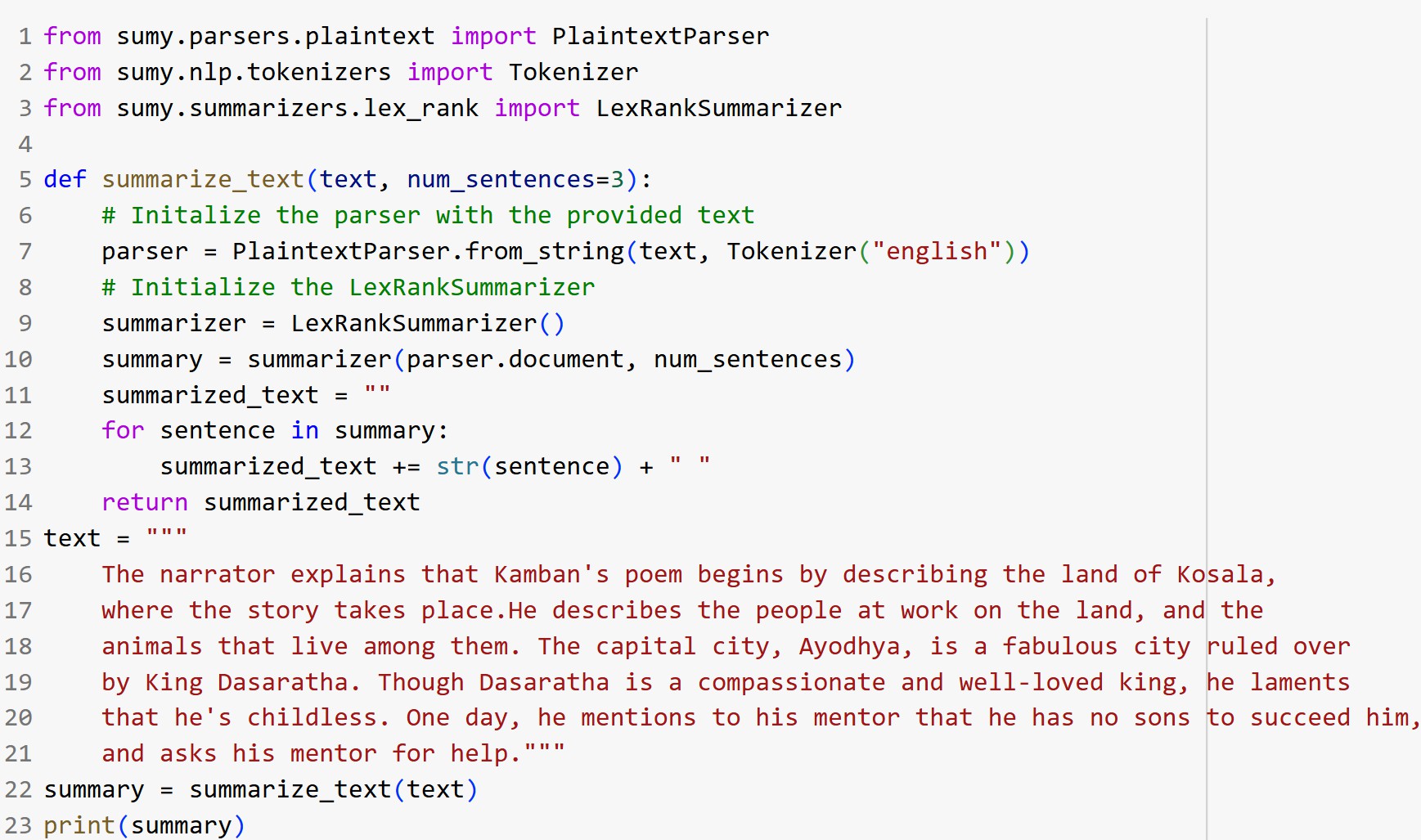
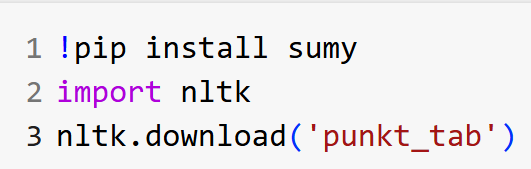
## Output:



(Web Mining - II) Practical No. 7

# DEPARTMENT OF INFORMATION TECHNOLOGY

|  |
| --- |
| 1. **AIM: Write a python program to summarize Text** 2. **DESCRIPTION:**   "Summarization of data in web mining" refers to the process of extracting key information and trends from large volumes of web data, condensing it into a meaningful representation that highlights the most important aspects of the information.   1. **CODE AND OUTPUT:**   **!pip install sumy import nltk**  **nltk.download('punkt\_tab')**  **from sumy.parsers.plaintext import PlaintextParser from sumy.nlp.tokenizers import Tokenizer**  **from sumy.summarizers.lex\_rank import LexRankSummarizer**  **def summarize\_text(text, num\_sentences=3):**  **# Initialize the parser with the provided text**  **parser = PlaintextParser.from\_string(text, Tokenizer("english")) # Initialize the LexRankSummarizer**  **summarizer = LexRankSummarizer()**  **summary = summarizer(parser.document, num\_sentences)**  **summarized\_text = ""** |



for sentence in summary:

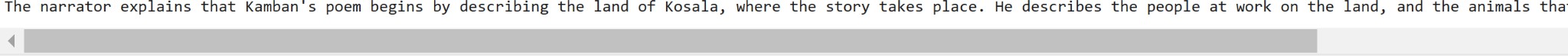
summarized\_text += str(sentence) + " " return summarized\_text

text = """

The narrator explains that Kamban's poem begins by describing the land of Kosala, where the story takes place.He describes the people at work on the land, and the animals that live among them. The capital city, Ayodhya, is a fabulous city ruled

over by King Dasaratha. Though Dasaratha is a compassionate and well-loved king, he laments that he's childless. One day, he mentions to his mentor that he has no sons to succeed him,

and asks his mentor for help.""" summary = summarize\_text(text) print(summary)



**Output:**



(Web Mining - II) Practical No. 8

**DEPARTMENT OF INFORMATION TECHNOLOGY**

|  |
| --- |
| 1. **AIM: Write a python program to implement a Recommender System.** 2. **DESCRIPTION:**   A recommender system suggests content or products based on your previous interactions, making your experience more personalized and relevant by utilizing user interaction data.   1. **CODE AND OUTPUT:**   **!pip install pandas scikit-surprise import pandas as pd**  **from surprise import Dataset,Reader from surprise import SVD**  **from surprise import accuracy**  **from surprise.model\_selection import train\_test\_split**  **data = {**  **'user\_id': [1,1,1,2,2,3,3,3,4,4],**  **'item\_id': ['A','B','C','A','C','A','B','D','B','C'], 'rating' : [5,3,4,4,8,6,1,2,4,1]**  **}**  **df = pd.DataFrame(data)**  **reader = Reader(rating\_scale=(1,5))**  **dataset = Dataset.load\_from\_df(df[['user\_id','item\_id','rating']],reader)** |

|  |
| --- |
| **trainset, testset = train\_test\_split(dataset, test\_size=0.2) # test\_size=0.2 means testing only 20% data**  **model = SVD() model.fit(trainset)**  **predictions = model.test(testset) rmse = accuracy.rmse(predictions) print(f"RMSE: {rmse}")**  **pred = model.predict(3,'B')**  **print(f"Prediction for user 1, item A: {pred.est}")** |
| **Output:** |

