Bhartiya Vidya Bhavan's
Sardar Patel Institute of Technology
Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

Name:	Utsav Avaiya, Ayush Bodade
UID:	2021300005, 2021300015
Experiment No.	7

AIM:	Demonstrate the behavior of Web Crawlers/spiders
Program 1	
PROBLEM STATEMENT :	Create a table weblink that has the seed link as given below which is the input to the crawler. Create a crawler that crawls the pages, add the rank, 4/5 keywords per page Crawls to max depth of 10.
PROGRAM:	

```
import requests
from bs4 import BeautifulSoup
import sqlite3
from urllib.parse import urljoin, urlparse

# Define the seed URL and maximum depth
seed_url = "https://en.wikipedia.org/wiki/Web_crawler"
max_depth = 5

# Initialize the database with the seed link
conn = sqlite3.connect("webcrawler.db")
cursor = conn.cursor()

# Create a table for storing crawled data
cursor.execute('''
CREATE TABLE IF NOT EXISTS weblink (
id INTEGER PRIMARY KEY,
url TEXT,
```

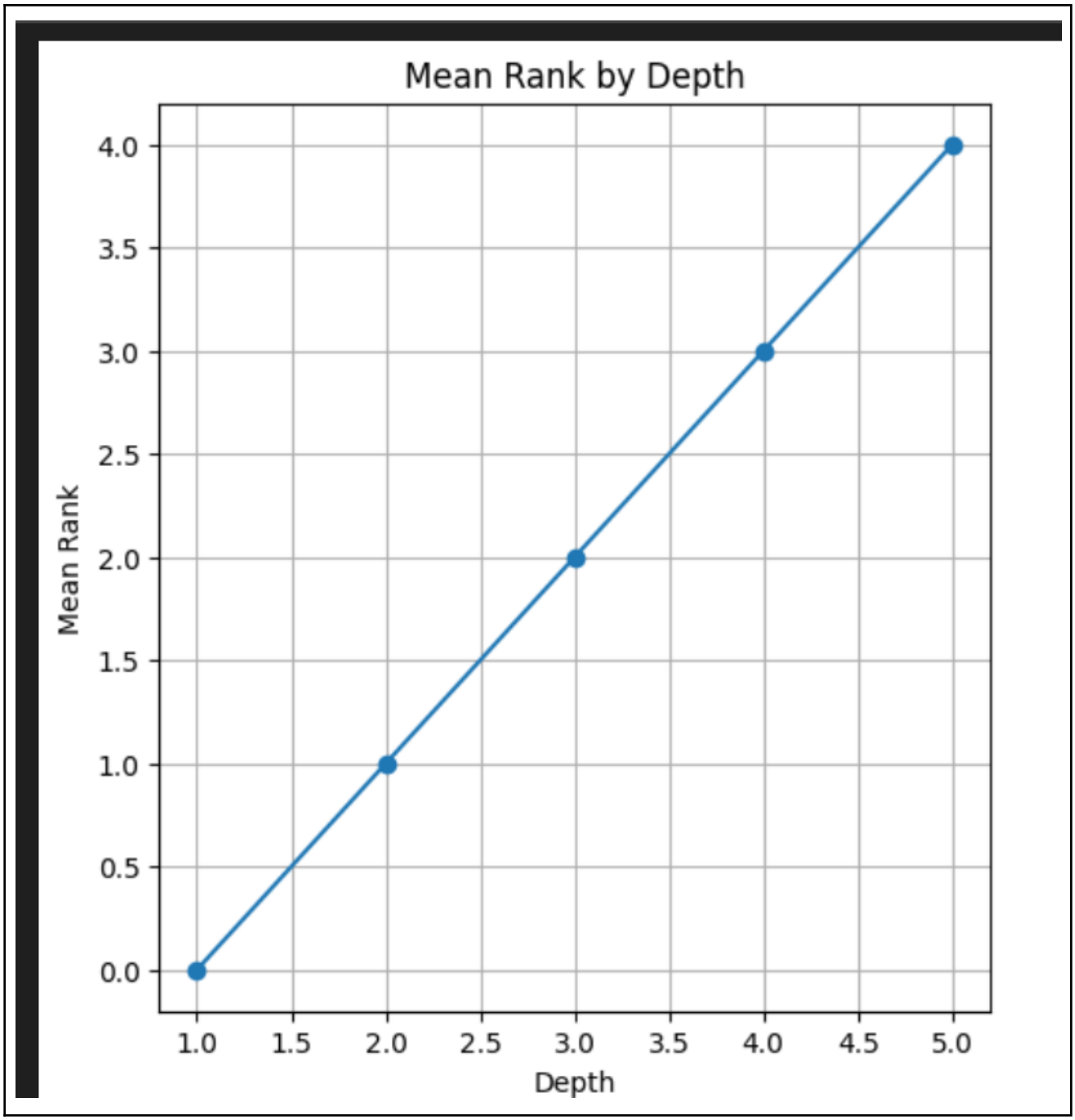
```
rank INTEGER,
keywords TEXT,
depth INTEGER
)
''')

# Maintain a set to track visited URLs and avoid
duplicates
visited_urls = set()

# Function to extract keywords from the page content (you
can implement your own method)
def extract_keywords(soup):
# This is a placeholder. Implement a method to extract
keywords from the page content.
return "HTTP, webpage, policies"

# Function to crawl a URL and store data in the database
def crawl_url(url, depth, rank, parent_keywords):
if depth <= max_depth and url not in visited_urls:
try:
response = requests.get(url)
if response.status_code == 200:
soup = BeautifulSoup(response.text, 'html.parser')

# Extract keywords from the page content (you can use more
advanced techniques for this)
page_keywords = extract_keywords(soup)

# Insert data into the database
cursor.execute("INSERT INTO weblink (url, rank, keywords,
depth) VALUES (?, ?, ?, ?)",
(url, rank, page_keywords, depth))
conn.commit()

# Mark the URL as visited to avoid duplicates
visited_urls.add(url)

# Crawl links on this page
```

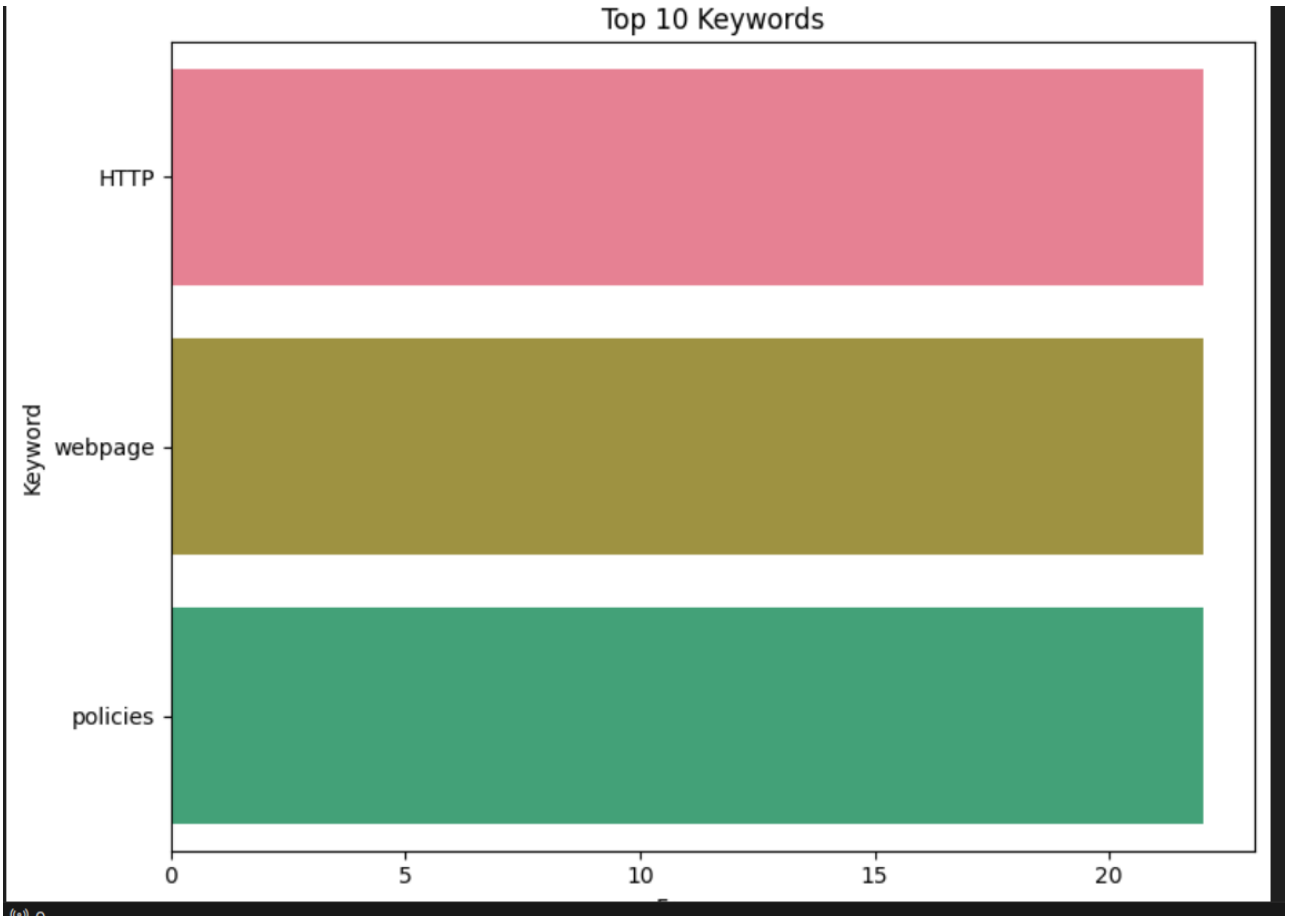
```
links = soup.find_all('a')
for link in links:
next_url = link.get('href')
if next_url:
# Make the URL absolute by joining it with the base URL
next_url = urljoin(url, next_url)

# Parse the URL to filter out unwanted links (e.g.,
external links)
parsed_url = urlparse(next_url)
if parsed_url.netloc == urlparse(seed_url).netloc:
crawl_url(next_url, depth + 1, rank + 1, page_keywords)

except Exception as e:
print(f"Error crawling URL {url}: {e}")

# Start the crawl with the seed URL
crawl_url(seed_url, 1, 0, "seed keywords")

# Close the database connection
conn.close()
```

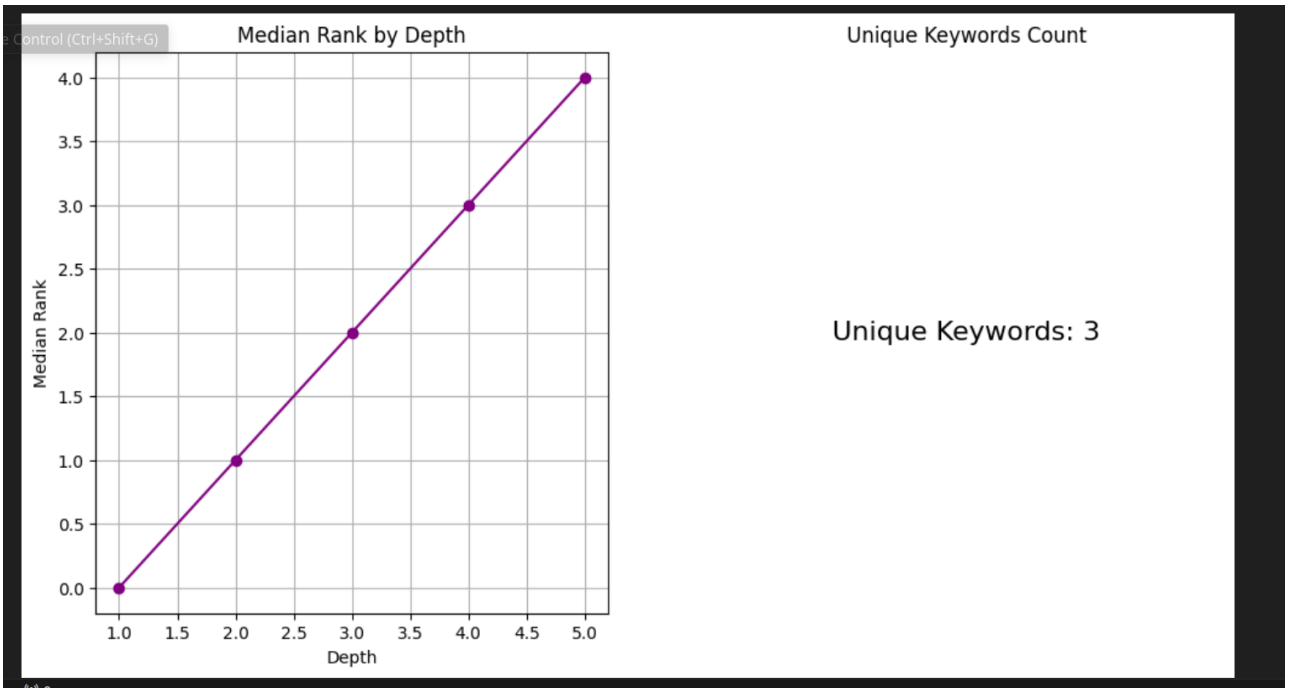
RESULT:

```
tl-webcrawler/interact.py
Crawled URLs and Depths:
URL: https://en.wikipedia.org/wiki/Web_crawler, Depth: 1

URL: https://donate.wikimedia.org/wiki/Special:FundraiserRedirector?utm_source=donate&utm_medium=sidebar&utm_campaign=C13_en.
wikipedia.org&uselang=en, Depth: 2

URL: https://donate.wikimedia.org/wiki/Special:LandingCheck?landing_page=Tax_deductibility&basic=true&language=en, Depth: 3

URL: https://donate.wikimedia.org/?uselang=en&utm_medium=donatewiki_page&utm_campaign=donate_now_btn&utm_source=Tax_deductibi
lity, Depth: 4

URL: https://donate.wikimedia.org/wiki/Special:LandingCheck?landing_page=Tax_deductibility&basic=true&language=en, Depth: 5

URL: https://donate.wikimedia.org/?uselang=en&utm_medium=donatewiki_page&utm_campaign=donate_now_btn&utm_source=Tax_deductibi
lity, Depth: 6

URL: https://donate.wikimedia.org/wiki/Special:LandingCheck?landing_page=Tax_deductibility&basic=true&language=en, Depth: 7

URL: https://donate.wikimedia.org/?uselang=en&utm_medium=donatewiki_page&utm_campaign=donate_now_btn&utm_source=Tax_deductibi
lity, Depth: 8

URL: https://donate.wikimedia.org/wiki/Special:LandingCheck?landing_page=Tax_deductibility&basic=true&language=en, Depth: 9
```

```
Average depth of crawled URLs: 4.91

No duplicate URLs found.

URLs crawled at depth 5:
https://en.wikipedia.org/wiki/Wikipedia:Contents
https://en.wikipedia.org/wiki/Portal:Current_events
https://en.wikipedia.org/wiki/Special:Random
https://en.wikipedia.org/wiki/Wikipedia:About
https://en.wikipedia.org/wiki/Wikipedia:Contact_us
https://en.wikipedia.org/wiki/Help:Contents
https://en.wikipedia.org/wiki/Help:Introduction
https://en.wikipedia.org/wiki/Wikipedia:Community_portal
https://en.wikipedia.org/wiki/Special:RecentChanges
https://en.wikipedia.org/wiki/Wikipedia:File_upload_wizard
https://en.wikipedia.org/wiki/Special:Search
https://en.wikipedia.org/w/index.php?title=Special:CreateAccount&returnto=Main+Page
https://en.wikipedia.org/w/index.php?title=Special:UserLogin&returnto=Main+Page
https://en.wikipedia.org/wiki/Special:MyContributions
https://en.wikipedia.org/wiki/Special:MyTalk
https://en.wikipedia.org/wiki/Talk:Main_Page
https://en.wikipedia.org/w/index.php?title=Main_Page&action=edit
https://en.wikipedia.org/w/index.php?title=Main_Page&action=history
```


Mean Rank by Depth
Mean Rank
Depth
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0





CONCLUSION:	Understood how to make web crawlers and implemented it using python,

	after which analysis was done to get insights from the database created.