Programme	:	**B.Tech** Semester : **Win Sem 21-22**
Course	:	**Web Mining Lab** Code : **CSE3024**
Faculty	:	**Dr.Bhuvaneswari A** Slot : **L7+L8**
Date	:	**14-01-2022** Marks : **10 Points**

Vaibhav Agarwal

19BCE1413

1. Implement crawlers which take as input a url, a search word and maximum number of pages to be searched and returns as output all the web pages it searched till it found the search word on a web page or return failure. (5 Marks)
    a. Breadth-First-Search
    b. Depth-First-Search

1A) BFS

```python
from bs4 import BeautifulSoup
import requests
```

[1] Python

```python
from urllib.request import urljoin
from urllib.request import urlparse
```

[2] Python

```python
links_intern = set()
links_extern = set()
input_url=input("Enter Link:")
depth=1
```

[3] Python

```
... Enter Link:https://vit.ac.in
```

```python
def level_crawler(input_url):
    temp_urls=set()
    current_url_domain=urlparse(input_url).netloc
    beautiful_soup_object = BeautifulSoup(
requests.get(input_url).content, "lxml")
    for anchor in beautiful_soup_object.findAll("a"):
        href = anchor.attrs.get("href")
        if(href!="" or href!=None):
            href=urljoin(input_url, href)
            href_parsed=urlparse(href)
            href=href_parsed.scheme
            href+="://"
            href+=href_parsed.netloc
            href+=href_parsed.path
            final_parsed_href=urlparse(href)
            is_valid=bool(final_parsed_href.scheme) and bool(
final_parsed_href.netloc)
            if is_valid:
                if current_url_domain not in href and href not in links_extern:
                    print("External - {}".format(href))
                    links_extern.add(href)
                if current_url_domain in href and href not in links_intern:
                    print("Internal - {}".format(href))
                    links_intern.add(href)
                    temp_urls.add(href)
    return temp_urls
```

[4] Python

```python
    if(depth == 0):
        print("Intern - {}".format(input_url))
    elif(depth == 1):
        level_crawler(input_url)
    else:
        #We have used a BFS approach
        # considering The structure as a tree.
        # It uses queue based approach to traverse links upto a particular depth
        queue = []
        queue.append(input_url)
        for j in range(depth):
            for count in range(len(queue)):
                url = queue.pop(0)
                urls = level_crawler(url)
                for i in urls:
                    queue.append(i)
```
[5] Python

```
Internal - https://vit.ac.in
Internal - https://viteee.vit.ac.in/
Internal - http://chennai.vit.ac.in/
External - https://vitap.ac.in/
External - https://vitbhopal.ac.in/
Internal - https://vit.ac.in/
Internal - https://vit.ac.in/about-vit
Internal - https://vit.ac.in/about/vision-mission
Internal - https://vit.ac.in/vit-milestones
Internal - https://vit.ac.in/about/leadership
Internal - https://vit.ac.in/governance
Internal - https://vit.ac.in/about/administrative-offices
Internal - https://vit.ac.in/about/infrastructure
Internal - https://vit.ac.in/about/ranking-and-accreditation
Internal - https://vit.ac.in/about/sustainability
Internal - https://vit.ac.in/true-green
Internal - https://vit.ac.in/about/community-outreach
Internal - https://vit.ac.in/about/communityradio
Internal - https://vit.ac.in/all-news-archieved
Internal - https://vit.ac.in/all-events
Internal - https://vit.ac.in/national-institutional-ranking-framework-nirf
Internal - https://vit.ac.in/mhrdugc
Internal - http://careers.vit.ac.in/
Internal - https://vit.ac.in/about/news-letter
Internal - https://vit.ac.in/academics/home
```

show more (open the raw output data in a text editor) ...

show more (open the raw output data in a text editor) ...

```
External - https://www.vitaa.org/
Internal - https://vit.ac.in/contactus
Internal - https://vit.ac.in/guesthouse/
External - https://www.facebook.com/vituniversity/
Internal - https://campustour.vit.ac.in
```

1B) DFS

```python
import requests
from bs4 import BeautifulSoup
```
[6] Python

```python
def dfs(base, path,visited,max_depth=3, depth=0):
    if depth < max_depth:
        try:
            soup = BeautifulSoup(requests.get(base + path).text, "html.parser")
            for link in soup.find_all("a"):
                href = link.get("href")
                if href not in visited:
                    visited.add(href)
                    print(f"at depth {depth}: {href}")
                    if href.startswith("http"):
                        dfs(href, "", visited,max_depth-1, depth + 1)
                    else:
                        dfs(base, href,visited,max_depth-1, depth + 1)
        except:
            pass
```
[7] Python

```python
link="https://www.geeksforgeeks.org"
dfs(link, "", set([link]))
```
[8] Python

```
at depth 0: #main
at depth 1: https://www.geeksforgeeks.org/
at depth 1: https://www.geeksforgeeks.org/must-do-coding-questions-for-product-based-companies/?ref=ghm
at depth 1: https://practice.geeksforgeeks.org/topic-tags/
at depth 1: https://practice.geeksforgeeks.org/company-tags
at depth 1: https://www.geeksforgeeks.org/analysis-of-algorithms-set-1-asymptotic-analysis/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/analysis-of-algorithms-set-2-asymptotic-analysis/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/analysis-of-algorithms-set-3asymptotic-notations/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/analysis-of-algorithems-little-o-and-little-omega-notations/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/lower-and-upper-bound-theory/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/analysis-of-algorithms-set-4-analysis-of-loops/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/analysis-algorithm-set-4-master-method-solving-recurrences/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/analysis-algorithm-set-5-amortized-analysis-introduction/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/g-fact-86/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/pseudo-polynomial-in-algorithms/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/polynomial-time-approximation-scheme/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/a-time-complexity-question/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/searching-algorithms/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/sorting-algorithms/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/algorithms-gq/pattern-searching/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/geometric-algorithms/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/mathematical-algorithms/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/bitwise-algorithms/?ref=ghm
at depth 1: https://www.geeksforgeeks.org/randomized-algorithms/?ref=ghm
```

```
show more (open the raw output data in a text editor) ...

at depth 1: https://www.geeksforgeeks.org/careers/?job_type=1&ref=footer
at depth 1: https://www.geeksforgeeks.org/videos/?ref=footer
at depth 1: https://www.geeksforgeeks.org/copyright-information/
at depth 1: https://www.geeksforgeeks.org/cookie-policy/
at depth 1: https://www.geeksforgeeks.org/privacy-policy/
```

2 Google API

```python
import requests
from bs4 import BeautifulSoup
import re
```
[1] ✓ 0.8s Python

```python
root_URL = "https://www.vit.ac.in/"
```
[2] ✓ 0.2s Python

```python
response = requests.get(root_URL)
```
[3] ✓ 2.4s Python

```python
root_page = BeautifulSoup(response.content, 'html.parser')
```
[4] ✓ 0.5s Python

```python
for data in root_page.find_all("p"):
    source=data.get_text()
    for i in re.findall(r'[\+\(]?[1-9][0-9 .\-\(\)]{8,}[0-9]',source):
        print(i)
```
[5] ✓ 0.2s Python