

Experiment-4

Depth First Search (DFS) and Breadth First Search (BFS)

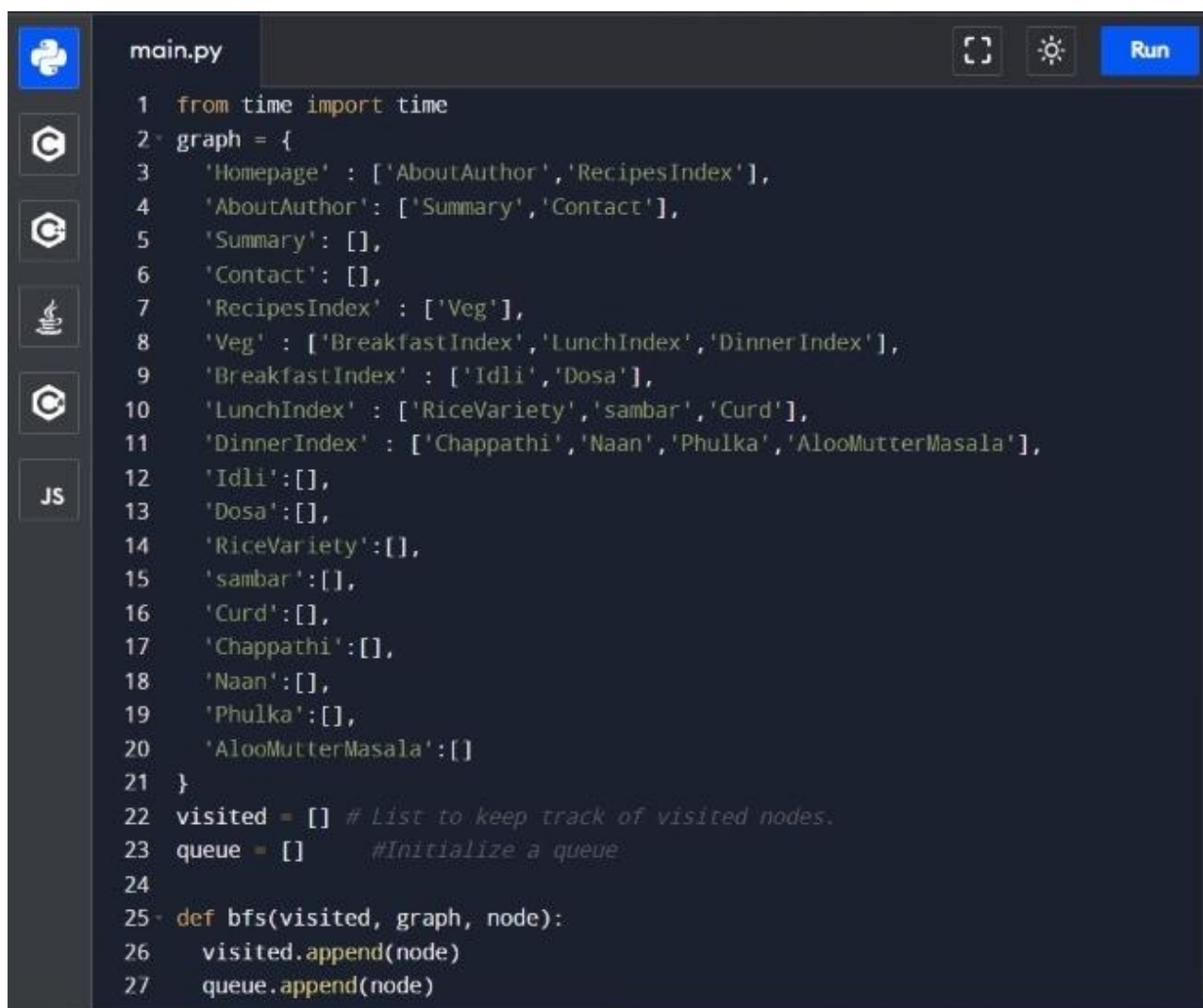
Team AI4Life
Sai Mohit Ambekar (137)
Sadekar Adesh(141)
Kapuluru Srinivasulu(142)
Praneet Botke(149)
Aayushi Goenka(151)
Sonia Raja(152)

Aim: To implement and analyze DFS and BFS for an application

Problem Statement & Solution: A web crawler bot is something to search the World Wide Web automatically for Web indexing. The problem here is to show how the DFS and BFS traverse through a simple web page. The idea is to start from source page and follow all links from source and keep doing same using DFS and BFS.

Code:

- BFS

A screenshot of a code editor window titled 'main.py'. The editor has a dark theme and a sidebar on the left with icons for Python, a web browser, a terminal, and a file explorer. The code is a Python script implementing Breadth First Search (BFS) on a graph representing a website structure. The graph is defined as a dictionary where keys are page names and values are lists of linked pages. The BFS function is defined to take a visited list, the graph, and a starting node, and it appends the node to both lists. The code is as follows:

```
1 from time import time
2 graph = {
3     'Homepage' : ['AboutAuthor','RecipesIndex'],
4     'AboutAuthor': ['Summary','Contact'],
5     'Summary': [],
6     'Contact': [],
7     'RecipesIndex' : ['Veg'],
8     'Veg' : ['BreakfastIndex','LunchIndex','DinnerIndex'],
9     'BreakfastIndex' : ['Idli','Dosa'],
10    'LunchIndex' : ['RiceVariety','sambar','Curd'],
11    'DinnerIndex' : ['Chappathi','Naan','Phulka','AlooMutterMasala'],
12    'Idli':[],
13    'Dosa':[],
14    'RiceVariety':[],
15    'sambar':[],
16    'Curd':[],
17    'Chappathi':[],
18    'Naan':[],
19    'Phulka':[],
20    'AlooMutterMasala':[]
21 }
22 visited = [] # List to keep track of visited nodes.
23 queue = [] #Initialize a queue
24
25 def bfs(visited, graph, node):
26     visited.append(node)
27     queue.append(node)
```


Output:

```
Shell Clear
Homepage
AboutAuthor
RecipesIndex
Summary
Contact
Veg
BreakfastIndex
LunchIndex
DinnerIndex
Idli
Dosa
RiceVariety
sambar
Curd
Chappathi
Naan
Phulka
AlooMutterMasala
Time for BFS : 0.00014591217041015625 seconds
>
```

- DFS

```
main.py Run
1 from time import time
2 graph = {
3     'Homepage' : ['AboutAuthor', 'RecipesIndex'],
4     'AboutAuthor': ['Summary', 'Contact'],
5     'Summary': [],
6     'Contact': [],
7     'RecipesIndex' : ['Veg'],
8     'Veg' : ['BreakfastIndex', 'LunchIndex', 'DinnerIndex'],
9     'BreakfastIndex' : ['Idli', 'Dosa'],
10    'LunchIndex' : ['RiceVariety', 'sambar', 'Curd'],
11    'DinnerIndex' : ['Chappathi', 'Naan', 'Phulka', 'AlooMutterMasala'],
12    'Idli':[],
13    'Dosa':[],
14    'RiceVariety':[],
15    'sambar':[],
16    'Curd':[],
17    'Chappathi':[],
18    'Naan':[],
19    'Phulka':[],
```

Output:



```
Shell
Homepage
AboutAuthor
Summary
Contact
RecipesIndex
Veg
BreakfastIndex
Idli
Dosa
LunchIndex
RiceVariety
sambar
Curd
DinnerIndex
Chappathi
Naan
Phulka
AlooMutterMasala
Time for DFS : 0.0001323223114013672 seconds
```

Result: The programs were run successfully. From the output, it is clear that time taken to traverse the nodes using DFS is better than BFS.