**EXPERIMENT 7**

**Date:**

**Objective:** Write a program to implement DFS using Python

**Software Used:** Python

**Theory:**

Depth-first search is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node (selecting some arbitrary node as the root node in the case of a graph) and explores as far as possible along each branch before backtracking. So the basic idea is to start from the root or any arbitrary node and mark the node and move to the adjacent unmarked node and continue this loop until there is no unmarked adjacent node. Then backtrack and check for other unmarked nodes and traverse them. Finally, print the nodes in the path.

Create a recursive function that takes the index of the node and a visited array.

1. Mark the current node as visited and print the node.
2. Traverse all the adjacent and unmarked nodes and call the recursive function with the index of the adjacent node.

**Code:**

graph = {

'5' : ['3','7'],

'3' : ['2', '4'],

'7' : ['8'],

'2' : [],

'4' : ['8'],

'8' : []

}

visited = set() # Set to keep track of visited nodes of graph.

**def** **dfs**(visited, graph, node): #function for dfs

**if** node **not** **in** visited:

**print** (node)

visited.add(node)

**for** neighbour **in** graph[node]:

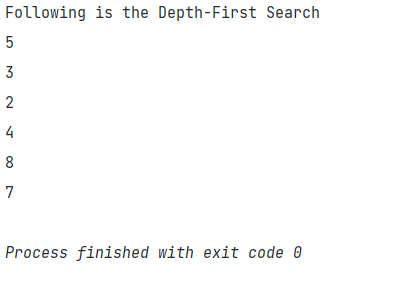
dfs(visited, graph, neighbour)

# Driver Code

**print**("Following is the Depth-First Search")

dfs(visited, graph, '5')

**OUTPUT:**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment Department of Computer Science & Engineering Amity University, Noida (UP) | | | | |
| Programme | B. Tech CSE | | Course Name | Artificial Intelligence |
| Course Code | [CSE401] | | Semester | 6 |
| Student Name | Krishna Sahu | | Enrollment No. | A2305219080 |
| Marking Criteria | | | | |
| Criteria | Total Marks | Marks Obtained | | Comments |
| Concept (A) | 2 |  | |  |
| Implementation (B) | 2 |  | |  |
| Performance (C) | 2 |  | |  |
| Total | 6 |  | |  |

**EXPERIMENT 8**

**Date:**

**Objective:** Tokenization of word and Sentences with the help of NLTK package

**Software Used:** Python

**Theory:**

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus comprehensive API documentation, NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project.

NLTK has been called “a wonderful tool for teaching, and working in, computational linguistics using Python,” and “an amazing library to play with natural language.”

**Code:**

import nltk

from nltk.tokenize import word\_tokenize

nltk.download('punkt')

text = "Dr. APJ Abdul Kalam is a famous name in the whole world. \

He is counted among the greatest scientists of the 21st century.\

 Even more, he becomes the 11th president of India and served his country.\

  He was the most valued person of the country as his contribution as a scie ntist and as a president is beyond compare.\

   Apart from that, his contribution to the ISRO (Ind ian Space Research Organization) is remarkable.\

    He headed many projects that contributed to the society also he was the one who helped in the development of Agni and Prithvi missiles.\

     For his involvement in the Nuclear power in India, he was known as Missile Man of India. And due to his contribution to the country, the government awarded him with the highest civilian award."

print('\n\nOutput of word tokenizer: \n', word\_tokenize(text))

from nltk.tokenize import sent\_tokenize

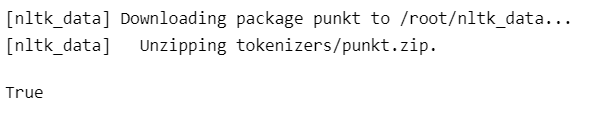
print("\n\nOutput of sentence tokenizer: \n", sent\_tokenize(text))

from nltk.tokenize import WordPunctTokenizer

wp = WordPunctTokenizer()

print('\n\nOutput of Punctuation Tokenizer: \n', wp.tokenize(text))

**OUTPUT:**





|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment Department of Computer Science & Engineering Amity University, Noida (UP) | | | | |
| Programme | B. Tech CSE | | Course Name | Artificial Intelligence |
| Course Code | [CSE401] | | Semester | 6 |
| Student Name | Krishna Sahu | | Enrollment No. | A2305219080 |
| Marking Criteria | | | | |
| Criteria | Total Marks | Marks Obtained | | Comments |
| Concept (A) | 2 |  | |  |
| Implementation (B) | 2 |  | |  |
| Performance (C) | 2 |  | |  |
| Total | 6 |  | |  |

**EXPERIMENT 9**

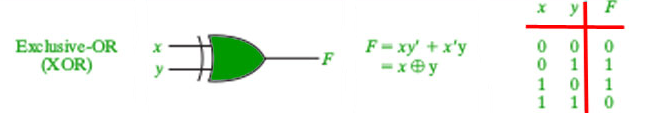
**Date:**

**Objective:** Design an XOR truth table using Python

**Software Used:** Python

**Theory:**

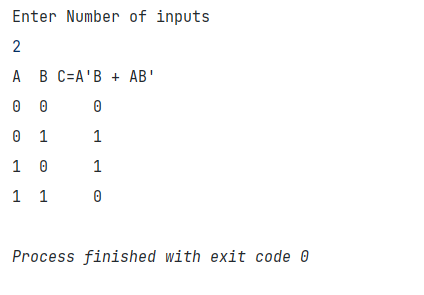
The XOR gate gives an output of 1 if either of the inputs is different, it gives 0 if they are the same. 



**Code:**

def decimalToBinary(n):  
 return bin(n).replace("0b", "")  
  
  
def xorTable(l):  
 for i in l:  
 for j in i:  
 if i.count('1') % 2 == 0:  
 output = '0'  
 else:  
 output = '1'  
 print(j, end=" ")  
 print(" ", output)  
  
  
print("Enter Number of inputs")  
n = int(input())  
if n == 2:  
 print("A B C=A\'B + AB\'")  
elif n == 3:  
 print("A B C D=ABC+A\'B\'C+AB\'C\'+A\'BC\'")  
  
l = []  
t = 2 \*\* n  
for i in range(t):  
 q = decimalToBinary(i)  
 q = str(q)  
 a = q.zfill(n)  
 l.append(a)  
  
# Driver\_Code  
if \_\_name\_\_ == '\_\_main\_\_':  
 xorTable(l)

OUTPUT:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment Department of Computer Science & Engineering Amity University, Noida (UP) | | | | |
| Programme | B. Tech CSE | | Course Name | Artificial Intelligence |
| Course Code | [CSE401] | | Semester | 6 |
| Student Name | Krishna Sahu | | Enrollment No. | A2305219080 |
| Marking Criteria | | | | |
| Criteria | Total Marks | Marks Obtained | | Comments |
| Concept (A) | 2 |  | |  |
| Implementation (B) | 2 |  | |  |
| Performance (C) | 2 |  | |  |
| Total | 6 |  | |  |