Experiment 8

Student Name: Ankita
UID: 22BCS17162
Branch: CSE
Section/Group:IOT_NTPP_602-A
Semester: 6th
Date of Performance:20-03-25
Subject Name: AP- 2
Subject Code: 22CSP-351

Aim:

a) Lowest Common Ancestor of a Binary Tree
b) Word Ladder
c) Number of Islands

Objective: To learn about Graphs.

Code:

a)

```
class Solution {
public:
TreeNode* lowestCommonAncestor(TreeNode* root, TreeNode* p, TreeNode* q) {
if (!root || root == p || root == q) {
return root;
}
TreeNode* leftLCA = lowestCommonAncestor(root->left, p, q);
TreeNode* rightLCA = lowestCommonAncestor(root->right, p, q);
if (leftLCA && rightLCA) {
return root;
}
return leftLCA ? leftLCA : rightLCA;
}
};
```

b)

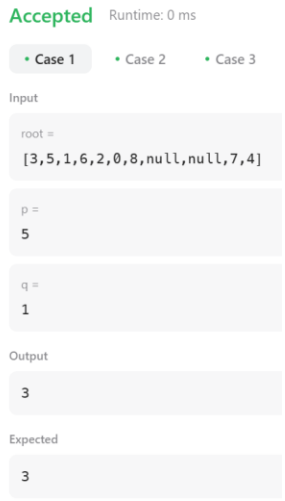
```
class Solution {
public:
int ladderLength(string beginWord, string endWord, vector<string>& wordList) {
unordered_set<string> wordSet(wordList.begin(), wordList.end());
if (wordSet.find(endWord) == wordSet.end()) return 0;
queue<pair<string, int>> q;
q.push({beginWord, 1});
while (!q.empty()) {
auto [word, steps] = q.front(); q.pop();
if (word == endWord) return steps;
for (int i = 0; i < word.size(); ++i) {
char originalChar = word[i];
for (char c = 'a'; c <= 'z'; ++c) {
if (c == originalChar) continue;
word[i] = c;
if (wordSet.find(word) != wordSet.end()) {
q.push({word, steps + 1});
wordSet.erase(word);}}
word[i] = originalChar;}}
return 0;}};
```

c)

```
class Solution {
public:
void dfs(vector<vector<char>>& grid, int i, int j) {
if (i < 0 || i >= grid.size() || j < 0 || j >= grid[0].size() || grid[i][j] == '0') {
return;}
grid[i][j] = '0';
dfs(grid, i + 1, j);
dfs(grid, i - 1, j);
dfs(grid, i, j + 1);
dfs(grid, i, j - 1);}
int numIslands(vector<vector<char>>& grid) {
int count = 0;
for (int i = 0; i < grid.size(); ++i) {
for (int j = 0; j < grid[0].size(); ++j) {
if (grid[i][j] == '1') {
++count;
dfs(grid, i, j); }}}
return count;
}};
```

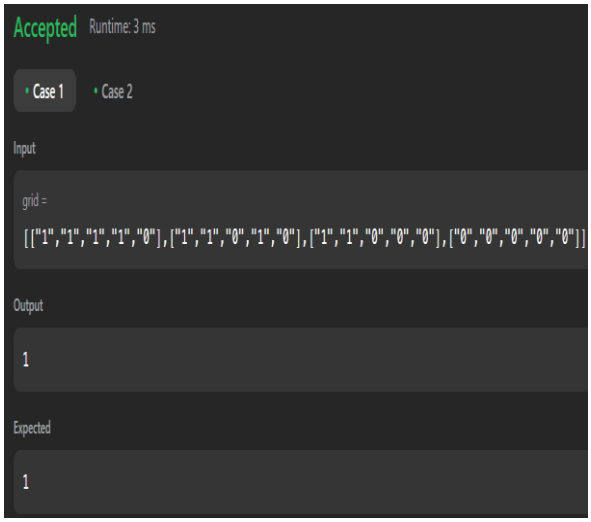
Output:

a)



b)

c)



Learning Outcomes:

a) Understand the concept of graph.
b) Learnt about different problem like word ladder, number of islands, etc.
c) Gain an understanding about the efficiency of graph.