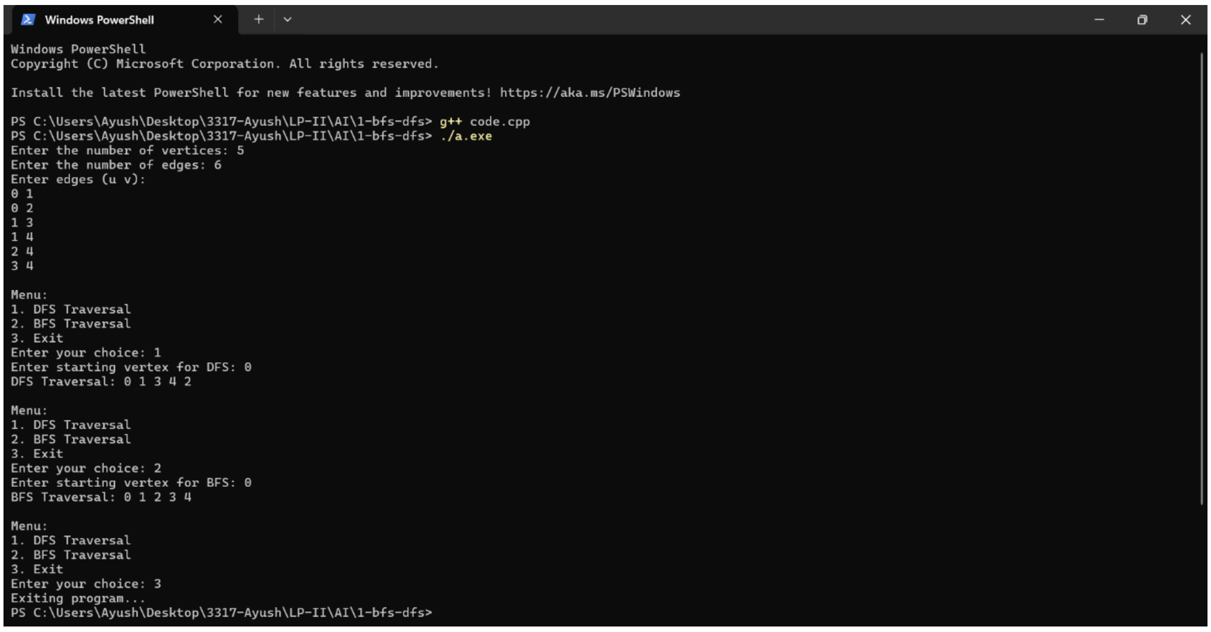
```cpp
/*
    Name : Ayush Pandey
    Roll No : 3317

    ASSIGNMENT-1
    Problem Statement :
        Implement Depth First Search algorithm and Breadth First Search algorithm.
        Use an undirected graph and develop a recursive algorithm for searching
        all the vertices of a graph or tree data structure.
*/

#include <iostream>
#include <vector>
#include <queue>

using namespace std;

class Graph {
    int vertices;
    vector<vector<int>> adjList;

public:
    Graph(int v) {
        vertices = v;
        adjList.resize(v);
    }

    void addEdge(int u, int v) {
        adjList[u].push_back(v);
        adjList[v].push_back(u);
    }

    void DFSUtil(int v, vector<bool>& visited) {
        cout << v << " ";
    visited[v] = true;

    for (int neighbor : adjList[v]) {
        if (!visited[neighbor]) {
            DFSUtil(neighbor, visited);
        }
    }
}

void DFS(int start) {
    vector<bool> visited(vertices, false);
    cout << "DFS Traversal: ";
    DFSUtil(start, visited);
    cout << endl;
}

void BFS(int start) {
    vector<bool> visited(vertices, false);
    queue<int> q;

    visited[start] = true;
    q.push(start);

    cout << "BFS Traversal: ";
    while (!q.empty()) {
        int node = q.front();
        q.pop();
        cout << node << " ";

        for (int neighbor : adjList[node]) {
            if (!visited[neighbor]) {
                visited[neighbor] = true;
                q.push(neighbor);
            }
```

```
                }
            }
            cout << endl;
        }
};

int main() {
    int vertices, edges;
    cout << "Enter the number of vertices: ";
    cin >> vertices;
    Graph g(vertices);

    cout << "Enter the number of edges: ";
    cin >> edges;

    cout << "Enter edges (u v):" << endl;
    for (int i = 0; i < edges; i++) {
        int u, v;
        cin >> u >> v;
        g.addEdge(u, v);
    }

    while (true) {
        cout << "\nMenu:\n";
        cout << "1. DFS Traversal\n";
        cout << "2. BFS Traversal\n";
        cout << "3. Exit\n";
        cout << "Enter your choice: ";

        int choice, start;
        cin >> choice;

        switch (choice) {
            case 1:
```

```cpp
                cout << "Enter starting vertex for DFS: ";
                cin >> start;
                g.DFS(start);
                break;
            case 2:
                cout << "Enter starting vertex for BFS: ";
                cin >> start;
                g.BFS(start);
                break;
            case 3:
                cout << "Exiting program...\n";
                return 0;
            default:
                cout << "Invalid choice! Please enter again.\n";
        }
    }

    return 0;
}
```



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Ayush\Desktop\3317-Ayush\LP-II\AI\1-bfs-dfs> g++ code.cpp
PS C:\Users\Ayush\Desktop\3317-Ayush\LP-II\AI\1-bfs-dfs> ./a.exe
Enter the number of vertices: 5
Enter the number of edges: 6
Enter edges (u v):
0 1
0 2
1 3
1 4
2 4
3 4

Menu:
1. DFS Traversal
2. BFS Traversal
3. Exit
Enter your choice: 1
Enter starting vertex for DFS: 0
DFS Traversal: 0 1 3 4 2

Menu:
1. DFS Traversal
2. BFS Traversal
3. Exit
Enter your choice: 2
Enter starting vertex for BFS: 0
BFS Traversal: 0 1 2 3 4

Menu:
1. DFS Traversal
2. BFS Traversal
3. Exit
Enter your choice: 3
Exiting program...
PS C:\Users\Ayush\Desktop\3317-Ayush\LP-II\AI\1-bfs-dfs>
```