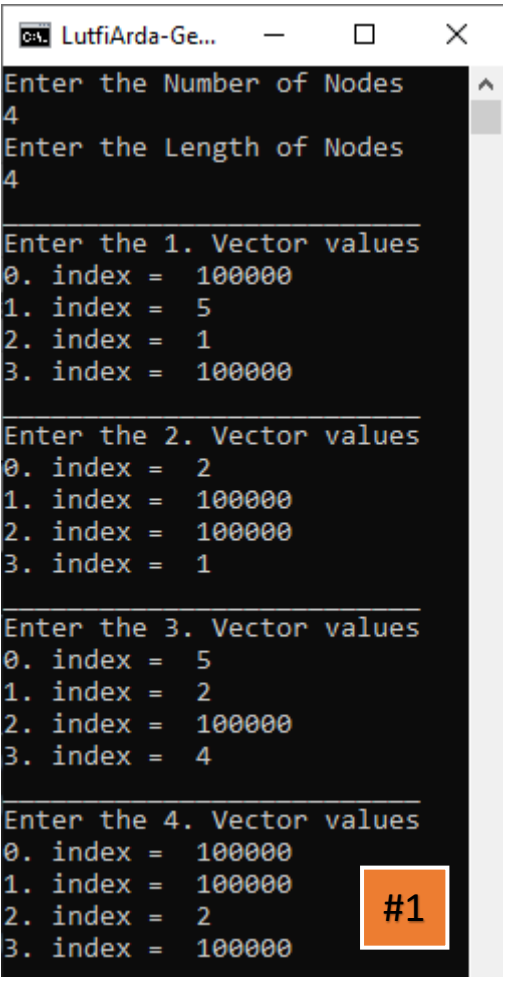
ASSIGNMENT #4 – Dynamic Programming-Floyd Algorithm

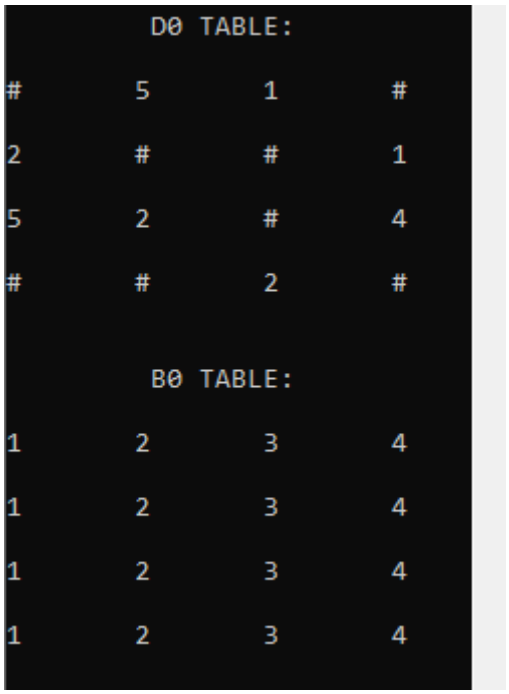
C Code of Floyd–Warshall Algorithm

```
#include <iostream>
#include <vector>
using namespace std;
void showvectortable(vector < vector<int> > floydvector) {
	for (int i = 0; i < floydvector.size(); i++) {
		for (int j = 0; j < floydvector[i].size(); j++) { // D table
			if (floydvector[i][j] == 100000) //  checks infinity
				cout << "#" << "\t";  // # for infinity
			else
				cout << floydvector[i][j] << "\t";
		}
		cout << "\n\n";
	}
}
int min(int x, int y) {
	if (x < y)
		return x;
	return y;
}
void floydAlgorithims(int node, vector < vector <int> > connection) {
	vector< vector<int> > d0(node, vector<int>(node)),
		d1(node, vector<int>(node)),
		b0(node, vector<int>(node)),
		b1(node, vector<int>(node));
	for (int i = 0; i < node; i++) {  // D and B table
		for (int j = 0; j < node; j++) {
			if (i != j)
			d0[i][j] = connection[i][j]; // Create D table
			else
			d0[i][j] = 100000;  // Infinity Value
			b0[i][j] = j + 1;  // B table
		}
	}
	for (int r = 0; r < node; r++) {
		cout << "\t D" << r << " TABLE:\n\n";
		showvectortable(d0);
		cout << "\n\t B" << r << " TABLE:\n\n";
		showvectortable(b0);
		for (int i = 0; i < node; i++) {
			for (int j = 0; j < node; j++) {
				if (d0[i][j] <= d0[i][r] + d0[r][j])  //
Create the B1
					b1[i][j] = b0[i][j];
				else
					b1[i][j] = b0[i][r]; // B1
				if (r == i || r == j) {
					d1[i][j] = d0[i][j];
					continue; // next iter
				}
				d1[i][j] = min(d0[i][j], d0[i][r] +
d0[r][j]);  // Floyd Algorithims Formula
			}
		}
		cout << "\n\n\n";
```

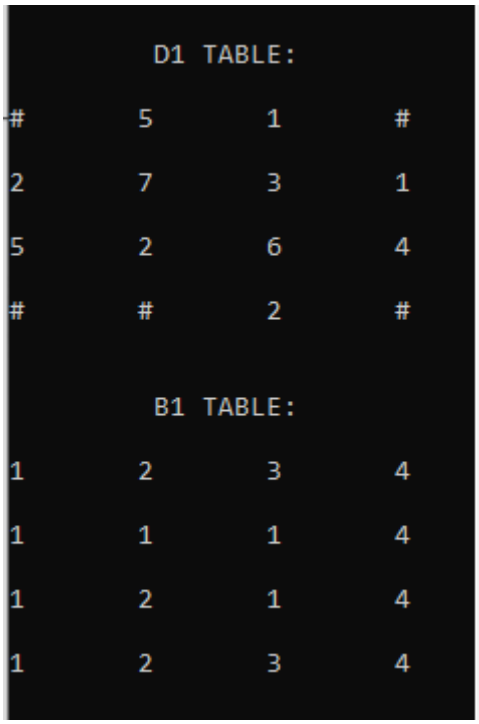
```
                d0 = d1;
                b0 = b1;
        }
        int target = 4;
        int start = 1;
        int i = 0;
        cout << "PATH = "<< start << " - ";
        while (true) {
                if (b0[i][target - 1] == target) {
                        cout << b0[i][target - 1];
                        break;
                }
                cout << b0[i][target - 1] << " - ";
                i = b0[i][target - 1] - 1;
        }
}
int main()
{
int length;
cout << "Enter the Number of Nodes " << "\n";
cin >> length;
int nodes;
cout << "Enter the Length of Nodes " << "\n";
cin >> nodes;
// {100000, 5, 1, 100000},{2, 100000, 100000, 1},{5, 2, 100000, 4},
{100000, 100000, 2, 100000}
vector<vector<int>> connection;
for(int i=0;i<nodes;++i) {
    cout << "Enter the " << i+1 <<  ". Vector values " << "\n";
        vector<int> row;
        for(int j=0;j<length;++j){
                int value;
                cout << j <<  ". index =  ";
                cin >> value;
                row.push_back(value);
        }
        //Push the row in matrix
        connection.push_back(row);
}
        cout << "\n\n";
        floydAlgorithims(nodes, connection);
}
```

Output of Program

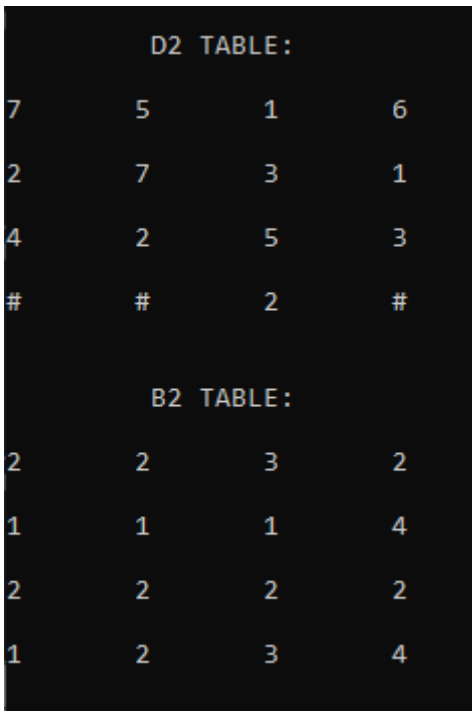




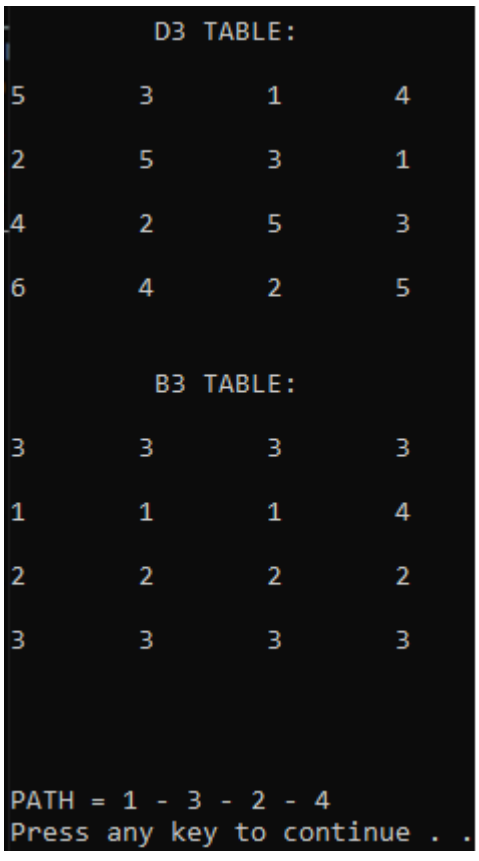
#2



#3



#4



#5

[Download the C Code File (until 29 December.)]