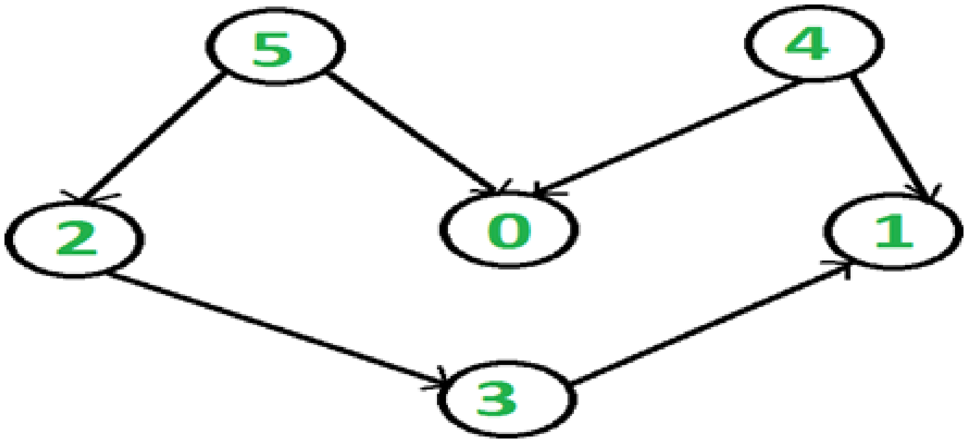
EXPERIMENT 2.2

Student Name: ARYAN GUPTA **UID: 20BCS6656**
Branch: CSE-AIML **Section/Group: 20AIML4-GRPB**
Semester: 5 **Date: 25/10/22**
Subject Name: Advance Programming **Subject Code: 20CSP-334_20AML-4_B**

AIM: Obtain the Topological ordering of vertices in a given digraph.

Task to be done: In this experiment we have to obtain Topological sorting of the below graph.



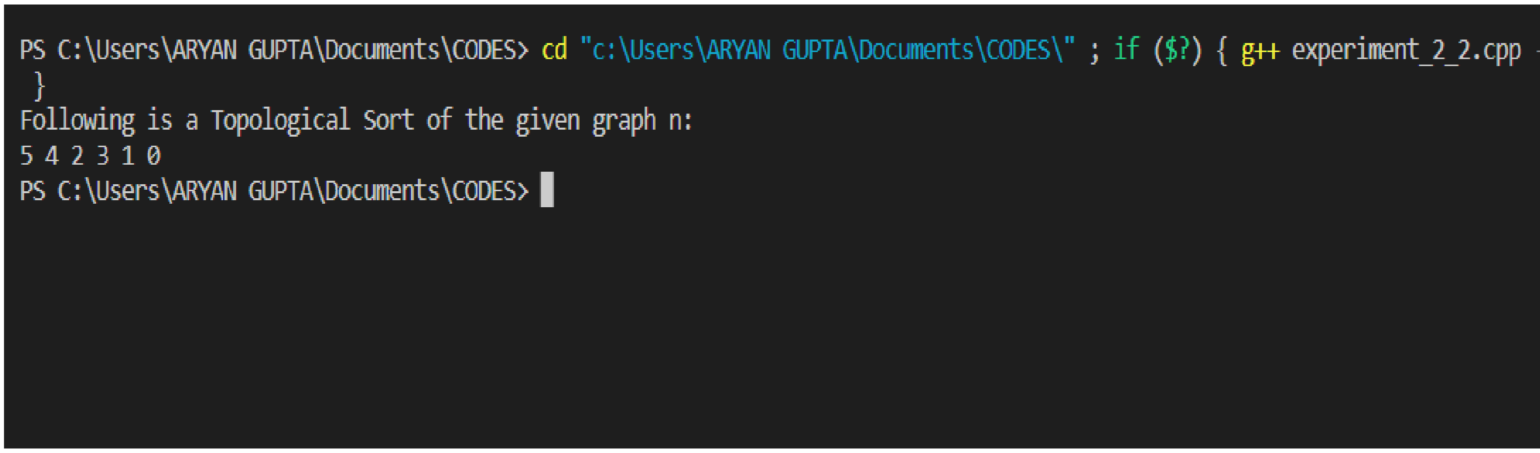
Algorithm to find Topological Sorting:
We recommend to first see the implementation of DFS. We can modify DFS to find Topological Sorting of a graph. In DFS, we start from a vertex, we first print it and then recursively call DFS for its adjacent vertices. In topological sorting, we use a temporary stack. We don't print the vertex immediately, we first recursively call topological sorting for all its adjacent vertices, then push it to a stack. Finally, print contents of the stack. Note that a vertex is pushed to stack only when all of its adjacent vertices (and their adjacent vertices and so on) are already in the stack.
Below image is an illustration of the above approach:

Code:

```
#include <iostream>
#include <list>
#include <stack>
using namespace std;

class Graph {
    int V;
    list<int>* adj;

    void topologicalSortUtil(int v, bool visited[], stack<int>& Stack);

public:
    Graph(int V);

    void addEdge(int v, int w);

    void topologicalSort();
};

Graph::Graph(int V)
{
    this->V = V;
    adj = new list<int>[V];
}

void Graph::addEdge(int v, int w)
{
    adj[v].push_back(w);
}

void Graph::topologicalSortUtil(int v, bool visited[],
                                stack<int>& Stack)
{

    visited[v] = true;

    list<int>::iterator i;
    for (i = adj[v].begin(); i != adj[v].end(); ++i)
        if (!visited[*i])
            topologicalSortUtil(*i, visited, Stack);
    Stack.push(v);
}

void Graph::topologicalSort()
{
    stack<int> Stack;

    bool* visited = new bool[V];
    for (int i = 0; i < V; i++)
        visited[i] = false;

    for (int i = 0; i < V; i++)
        if (visited[i] == false)
            topologicalSortUtil(i, visited, Stack);

    while (Stack.empty() == false) {
        cout << Stack.top() << " ";
        Stack.pop();
    }
}

int main()
{
    Graph g(6);
    g.addEdge(5, 2);
    g.addEdge(5, 0);
    g.addEdge(4, 0);
    g.addEdge(4, 1);
    g.addEdge(2, 3);
    g.addEdge(3, 1);

    cout << "Following is a Topological Sort of the given graph n: \n";
    g.topologicalSort();

    return 0;
}
```

Output:

1)



```
PS C:\Users\ARYAN GUPTA\Documents\CODES> cd "c:\Users\ARYAN GUPTA\Documents\CODES\" ; if ($?) { g++ experiment_2_2.cpp
 }
Following is a Topological Sort of the given graph n:
5 4 2 3 1 0
PS C:\Users\ARYAN GUPTA\Documents\CODES>
```

Learning Outcomes:

- Learnt about the Topological Sorting.
- Learnt about the Implementation of Topological Sorting in CPP.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			

-