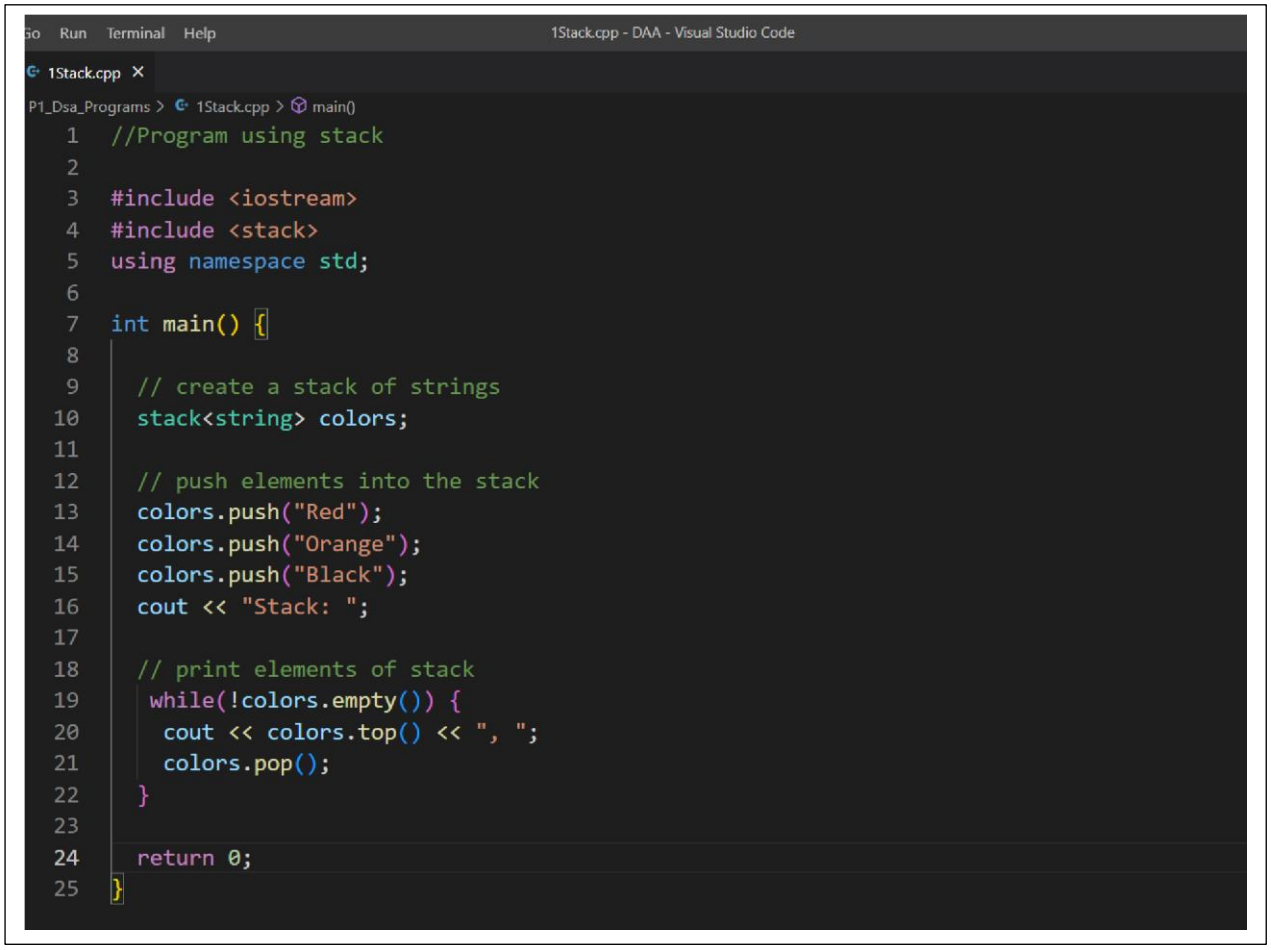
DAA Practical No 1

Aim: Write C/C++ code to implement concept of

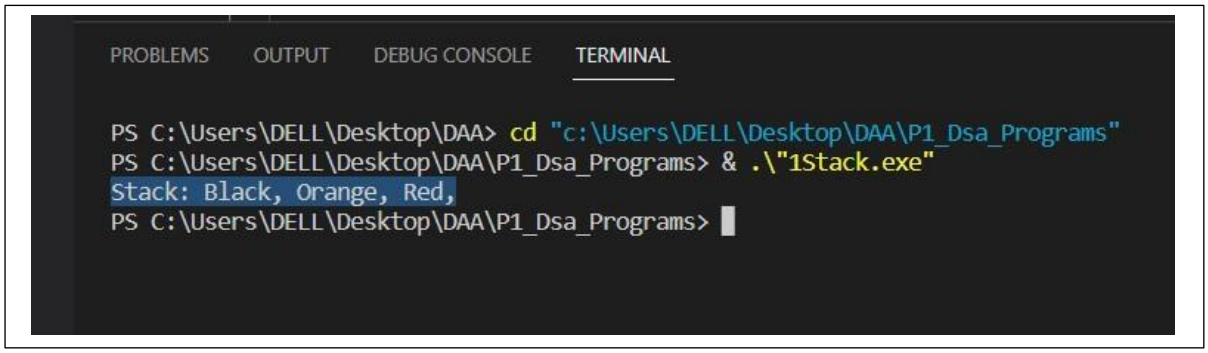
1) Stack
2) Queue
3) Linked List
4) Trees
5) Graphs

1)Stack

-- > Program:



Output:



2)Queue

-- >Program:

Go Run Terminal Help 2Queue.cpp - DAA - Visual Studio Code

2Queue.cpp

P1_Dsa_Programs > 2Queue.cpp > main()

```cpp
//We use the push() method to insert an element to the back of a queue

#include <iostream>
#include <queue>
using namespace std;

int main() {

  // create a queue of string
  queue<string> animals;

  // push elements into the queue
  animals.push("Cat");
  animals.push("Dog");

  cout << "Queue: ";

  // print elements of queue
  // loop until queue is empty
  while(!animals.empty()) {

    // print the element
    cout << animals.front() << ", ";

    // pop element from the queue
    animals.pop();
  }

  cout << endl;

  return 0;
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\DELL\Desktop\DAA> cd "c:\Users\DELL\Desktop\DAA\P1_Dsa_Programs"
PS C:\Users\DELL\Desktop\DAA\P1_Dsa_Programs> & .\"2Queue.exe"
Queue: Cat, Dog,
PS C:\Users\DELL\Desktop\DAA\P1_Dsa_Programs>
```

3)Linked List

-- >Program:

Go Run Terminal Help 3linkedlist.cpp - DAA - V

3linkedlist.cpp

P1_Dsa_Programs > 3linkedlist.cpp > main()

```cpp
// Linked list implementation in C++

#include <bits/stdc++.h>
#include <iostream>
using namespace std;

// Creating a node
class Node {
  public:
  int value;
  Node* next;
};

int main() {
  Node* head;
  Node* one = NULL;
  Node* two = NULL;
  Node* three = NULL;

  // allocate 3 nodes in the heap
  one = new Node();
  two = new Node();
  three = new Node();

  // Assign value values
  one->value = 1;
  two->value = 2;
  three->value = 3;

  // Connect nodes
  one->next = two;
  two->next = three;
  three->next = NULL;

  // print the linked list value
  head = one;
  while (head != NULL) {
    cout << head->value;
    head = head->next;
  }
}
```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\DELL\Desktop\DAA> cd "c:\Users\DELL\Desktop\DAA\P1_Dsa_Programs"
PS C:\Users\DELL\Desktop\DAA\P1_Dsa_Programs> & .\"3linkedlist.exe"
123
PS C:\Users\DELL\Desktop\DAA\P1_Dsa_Programs>
```

4)Trees

-- >Program:

Go Run Terminal Help 4tree.cpp - DAA - Visual Studio Code

4tree.cpp

P1_Dsa_Programs > 4tree.cpp > main()

```cpp
// Binary Tree in C++

#include <stdlib.h>

#include <iostream>

using namespace std;

struct node {
  int data;
  struct node *left;
  struct node *right;
};

// New node creation
struct node *newNode(int data) {
  struct node *node = (struct node *)malloc(sizeof(struct node));

  node->data = data;

  node->left = NULL;
  node->right = NULL;
  return (node);
}

// Traverse Preorder
void traversePreOrder(struct node *temp) {
  if (temp != NULL) {
    cout << " " << temp->data;
    traversePreOrder(temp->left);
    traversePreOrder(temp->right);
  }
}

```

```cpp
// Traverse Inorder
void traverseInOrder(struct node *temp) {
  if (temp != NULL) {
    traverseInOrder(temp->left);
    cout << " " << temp->data;
    traverseInOrder(temp->right);
  }
}

// Traverse Postorder
void traversePostOrder(struct node *temp) {
  if (temp != NULL) {
    traversePostOrder(temp->left);
    traversePostOrder(temp->right);
    cout << " " << temp->data;
  }
}

```

```
53  int main() {
54    struct node *root = newNode(1);
55    root->left = newNode(2);
56    root->right = newNode(3);
57    root->left->left = newNode(4);
58
59    cout << "preorder traversal: ";
60    traversePreOrder(root);
61    cout << "\nInorder traversal: ";
62    traverseInOrder(root);
63    cout << "\nPostorder traversal: ";
64    traversePostOrder(root);
65  }
```

Output:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

PS C:\Users\DELL\Desktop\DAA> cd "c:\Users\DELL\Desktop\DAA\P1_Dsa_Programs"
PS C:\Users\DELL\Desktop\DAA\P1_Dsa_Programs> & .\"4tree.exe"
preorder traversal:  1 2 4 3
Inorder traversal:  4 2 1 3
Postorder traversal:  4 2 3 1
PS C:\Users\DELL\Desktop\DAA\P1_Dsa_Programs>
```

5)Graphs

-->Program:

Go Run Terminal Help 5graphs.cpp - DAA - Visual Studio Code

5graphs.cpp

P1_Dsa_Programs > 5graphs.cpp > addEdge(int, int)

```cpp
// C++ program to print DFS traversal from a given vertex in a given graph

#include <bits/stdc++.h>
using namespace std;

// Graph class represents a directed graph using adjacency list representation
class Graph {
public:
    map<int, bool> visited;
    map<int, list<int> > adj;

    // function to add an edge to graph
    void addEdge(int v, int w);

    // DFS traversal of the vertices reachable from v
    void DFS(int v);
};
void Graph::addEdge(int v, int w)
{
    adj[v].push_back(w); // Add w to v's list.
}
void Graph::DFS(int v)
{
    // Mark the current node as visited and print it
    visited[v] = true;
    cout << v << " ";

    // Recur for all the vertices adjacent to this vertex
    list<int>::iterator i;
    for (i = adj[v].begin(); i != adj[v].end(); ++i)
        if (!visited[*i])
            DFS(*i);
}
```

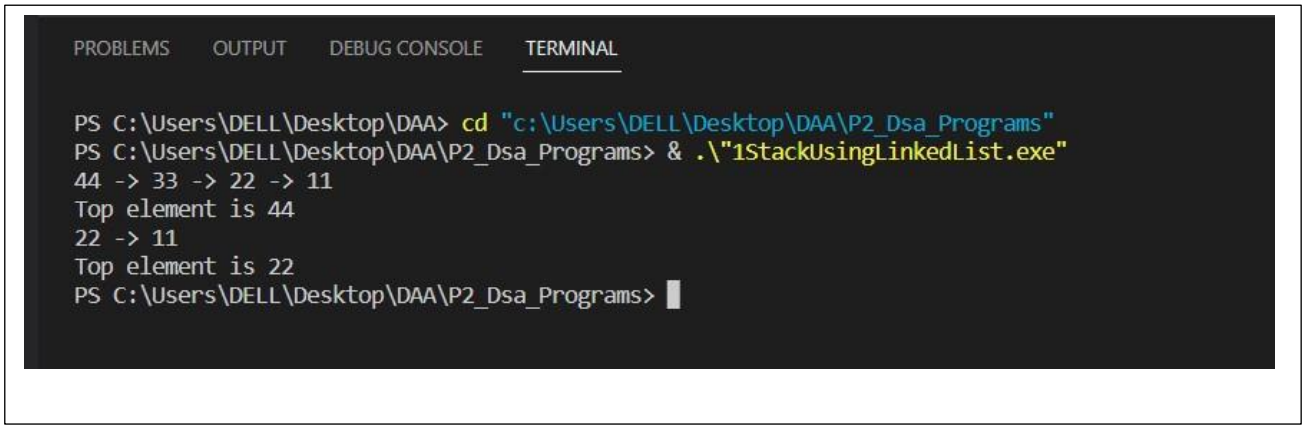
```cpp
int main()
{
    // Create a graph given in the above diagram
    Graph g;
    g.addEdge(0, 1);
    g.addEdge(0, 2);
    g.addEdge(1, 2);
    g.addEdge(2, 0);
    g.addEdge(2, 3);
    g.addEdge(3, 3);

    cout << "Following is Depth First Traversal"
            " (starting from vertex 2) \n";

    // Function call
    g.DFS(2);

    return 0;
}
```

Output:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PS C:\Users\DELL\Desktop\DAA> cd "c:\Users\DELL\Desktop\DAA\P2_Dsa_Programs"
PS C:\Users\DELL\Desktop\DAA\P2_Dsa_Programs> & .\"1StackUsingLinkedList.exe"
44 -> 33 -> 22 -> 11
Top element is 44
22 -> 11
Top element is 22
PS C:\Users\DELL\Desktop\DAA\P2_Dsa_Programs>