1. Write a C++ program to find all the strongly connected components in the given directed graph.

strongly_component.cpp - Code::Blocks 20.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

strongly_component.cpp | Task_scheduling.cpp

```cpp
#include <bits/stdc++.h>

using namespace std;

const int WHITE = 0;
const int GRAY = 1;
const int BLACK = 2;

vector<int> color;
vector<int> DescoveryTime;
vector<int> FinishTime;
vector<int> Ancestor;
stack<int> finishvectex;

int timeCount;

void DFS_VISIT(vector<vector<int>>& G, int u)
{
    timeCount++;
    DescoveryTime[u] = timeCount;
    color[u] = GRAY;

    for(int v=0; v<G[u].size(); v++)
    {
        if(G[u][v]==1 && color[v]==WHITE)
        {
            Ancestor[v] = u;
            DFS_VISIT(G, v);
        }
    }
    timeCount++;
    FinishTime[u] = timeCount;
    color[u] = BLACK;
    finishvectex.push(u);
}
```

"C:\Users\Golam Mostafa Rabby\OneDrive\B.Sc i...

```
{0 3 2 1 }
{4 6 5 }
{7 }

Process returned 0 (0x0)   execution time : 0.192 s
Press any key to continue.
```

```cpp
void DFS(vector<vector<int>>& G)
{
    int n = G.size();
    color.assign(n, WHITE);
    DescoveryTime.assign(n, 0);
    FinishTime.assign(n, 0);
    Ancestor.assign(n, -1);

    for(int u=0; u<n; u++)
    {
        if(color[u] == WHITE)
            DFS_VISIT(G, u);
    }
}


vector<vector<int>> Transpose(vector<vector<int>> & G)
{
    int n = G.size();
    vector<vector<int>> GT(n, vector<int>(n, 0));

    for(int u=0; u<n; u++)
    {
        for(int v=0; v<G[u].size(); v++)
        {
            if(G[u][v]==1)
                GT[v][u] = 1;
        }
    }
    return GT;
}

void DFS_VISIT_snd(vector<vector<int>>& G, int u, vector<int>& compone
{
    color[u] = GRAY;
    component.push_back(u);
```

C:\Users\Golam Mostafa Rabby\OneDrive\B.Sc in Duet\3rd year 1st semester\Sessiona... | C/C++ | Windows (CR+LF) | WINDOWS-1252 | Line 30, Col 6, Pos 563 | Insert | Read/Write | default

Search 11:16 PM 12/5/2024

strongly_component.cpp - Code::Blocks 20.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

strongly_component.cpp | Task_scheduling.cpp

```cpp
void DFS_VISIT_snd(vector<vector<int>>& G, int u, vector<int>& componer
{
    color[u] = GRAY;
    component.push_back(u);

    for(int v=0; v<G[u].size(); v++)
    {
        if(G[u][v]==1 && color[v]==WHITE)
        {
            DFS_VISIT_snd(G, v, component);
        }
    }
    color[u] = BLACK;
}

void dfs_fsccs(vector<vector<int>>& G)
{
    DFS(G);
    vector<vector<int>> GT = Transpose(G);
    int n = G.size();
    color.assign(n, WHITE);




    while(!finishvectex.empty())
    {
    int u = finishvectex.top();
    finishvectex.pop();
        if (color[u] == WHITE)
        {
            vector<int> component;
            DFS_VISIT_snd(GT, u, component);

            cout<<"{";
            for(auto i: component)
```

```cpp
    while(!finishvectex.empty())
    {
    int u = finishvectex.top();
    finishvectex.pop();
        if (color[u] == WHITE)
        {
            vector<int> component;
            DFS_VISIT_snd(GT, u, component);

            cout<<"{";
            for(auto i: component)
                cout<<i<<" ";
            cout<<"}"<<endl;
        }
    }
}

int main()
{
    vector<vector<int>> G=
    {
        {0,1,0,0, 0,0,0,0},
        {0,0,1,0, 0,0,0,0},
        {0,0,0,1, 1,0,0,0},
        {1,0,0,0, 0,0,0,0},

        {0,0,0,0, 0,1,0,0},
        {0,0,0,0, 0,0,1,0},
        {0,0,0,0, 1,0,0,1},
        {0,0,0,0, 0,0,0,0},
    };

    dfs_fsccs(G);

    return 0;
}
```

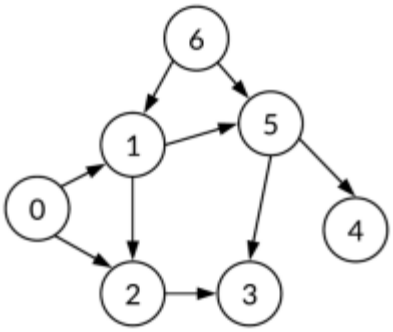
C:\Users\Golam Mostafa Rabby\OneDrive\B.Sc in Duet\3rd year 1st semester\Sessiona... | C/C++ | Windows (CR+LF) | WINDOWS-1252 | Line 30, Col 6, Pos 563 | Insert | Read/Write | default

Search 11:17 PM 12/5/2024

2. You are given a directed acyclic graph (DAG) representing N tasks, where:
   - Each node corresponds to a task.
   - A directed edge (u,v) indicates that task u must be completed before task v.

Write a C++ program to determine a valid order to complete all tasks. If multiple valid orders exist, any one of them is acceptable.

Unsorted graph



Topologcial_sort.cpp - Code::Blocks 20.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

strongly_component.cpp | Topologcial_sort.cpp

```cpp
#include <bits/stdc++.h>

using namespace std;

const int WHITE = 0;
const int GRAY = 1;
const int BLACK = 2;

vector<int> color;
vector<int> DescoveryTime;
vector<int> FinishTime;
vector<int> Ancestor;
vector<int> finishvectex;
int timeCount;

void DFS_VISIT(vector<vector<int>>& G, int u)
{
    timeCount++;
    DescoveryTime[u] = timeCount;
    color[u] = GRAY;

    for(int v=0; v<G[u].size(); v++)
    {
        if(G[u][v]==1 && color[v]==WHITE)
        {
            Ancestor[v] = u;
            DFS_VISIT(G, v);
        }
    }
    timeCount++;
    FinishTime[u] = timeCount;
    color[u] = BLACK;
    finishvectex.push_back(u);
}

void DFS(vector<vector<int>>& G)
{
    int n = G.size();
    color.assign(n, WHITE);
    DescoveryTime.assign(n, 0);
    FinishTime.assign(n, 0);
    Ancestor.assign(n, -1);
    finishvectex.clear();

    for(int u=0; u<n; u++)
    {
        if(color[u] == WHITE)
            DFS_VISIT(G, u);
    }
}
vector<int> Topological_sort(vector<vector<int>>& G)
{
    DFS(G);
    reverse(finishvectex.begin(), finishvectex.end());
    return finishvectex;
}
int main()
{
    vector<vector<int>> G=
    {  {0,1,1,0, 0,0,0},
       {0,0,1,0, 0,1,0},
       {0,0,0,1, 0,0,0},
       {0,0,0,0, 0,0,0},
       {0,0,0,0, 0,0,0},
       {0,0,0,1, 1,0,0},
       {0,1,0,0, 0,1,0}       };
    vector<int> SortedOrder = Topological_sort(G);
    for(int i: SortedOrder)
        cout<<i<<" ";
    cout<<endl;
    return 0;
}
```

"C:\Users\Golam Mostafa Rabby\OneDrive\B.Sc in...

```
6 0 1 5 4 2 3

Process returned 0 (0x0)   execution time : 0.185 s
Press any key to continue.
```

C:\Users\Golam Mostafa Rabby\OneDrive\B.Sc in Duet\3rd year 1st semester\Sessiona... | C/C++ | Windows (CR+LF) | WINDOWS-1252 | Line 23, Col 6, Pos 430 | Insert | Read/Write | default

Search 11:20 PM 12/5/2024