

**BÁCH KHOA E-LEARNING**

[Trang của tôi](#) / [Khoá học](#) / [Học kỳ II năm học 2021-2022 \(Semester 2 - Academic year 2021-2022\)](#)

/ [Chương Trình Chất Lượng Cao dạy bằng Tiếng Anh \(High-Quality training program\)](#)

/ [Khoa Khoa học và Kỹ thuật Máy tính \(Faculty of Computer Science and Engineering.\)](#) / [Khoa Học Máy Tính](#)

/ [Data Structures and Algorithms \(practice\) \(CO2004\) \\_Băng Ngọc Bảo Tâm \(CC\\_HK212\)](#) / Search + Hash + Graph / [Practice Exercises](#)

Câu hỏi **8**

Không hoàn thành

Chấm điểm của 1,00

Implement Depth-first search

```
Adjacency *DFS(int v);
```

where Adjacency is a structure to store list of number.

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

class Adjacency
{
private:
    list<int> adjList;
    int size;
public:
    Adjacency() {}
    Adjacency(int V) {}
    void push(int data)
    {
        adjList.push_back(data);
        size++;
    }
    void print()
    {
        for (auto const &i : adjList)
            cout << " -> " << i;
    }
    void printArray()
    {
        for (auto const &i : adjList)
            cout << i << " ";
    }
    int getSize() { return adjList.size(); }
    int getElement(int idx)
    {
        auto it = adjList.begin();
        advance(it, idx);
        return *it;
    }
};
```

And Graph is a structure to store a graph (see in your answer box)

For example:

Test	Result
<pre>int V = 8, visited = 0;  Graph g(V); Adjacency *arr; int edge[][2] = {{0,1}, {0,2}, {0,3}, {0,4}, {1,2}, {2,5}, {2,6}, {4,6}, {6,7}}; for(int i = 0; i &lt; 9; i++) {     g.addEdge(edge[i][0], edge[i][1]); }  // g.printGraph(); // cout &lt;&lt; endl; arr = g.DFS(visited); arr-&gt;printArray(); delete arr;</pre>	<pre>0 1 2 5 6 4 7 3</pre>

**Answer:** (penalty regime: 0 %)

Reset answer

```
1 class Graph
2 {
3 private:
4     int V;
5     Adjacency *adj;
6
7 public:
8     Graph(int V)
9     {
10         this->V = V;
11         adj = new Adjacency[V];
12     }
13
14     void addEdge(int v, int w)
15     {
16         adj[v].push(w);
17         adj[w].push(v);
18     }
19
20     void printGraph()
21     {
22         for (int v = 0; v < V; ++v)
23         {
24             cout << "\nAdjacency list of vertex " << v << "\nhead ";
25             adj[v].print();
26         }
27     }
28
29     Adjacency *DFS(int v)
30     {
31         // v is a vertex we start DFS
32     }
33 };
```

Kiểm tra

[◀ Practice Exercises](#)

Chuyển tới...

**Copyright 2007-2021 Trường Đại Học Bách Khoa - ĐHQG Tp.HCM. All Rights Reserved.**

Địa chỉ: Nhà A1- 268 Lý Thường Kiệt, Phường 14, Quận 10, Tp.HCM.

Email: [elearning@hcmut.edu.vn](mailto:elearning@hcmut.edu.vn)

Phát triển dựa trên hệ thống Moodle