AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY (AUST)
141 & 142, Love Road, Tejgaon Industrial Area, Dhaka-1208.

Department of Computer Science and Engineering

Program: Bachelor of Science in Computer Science and Engineering

Assignment Report

Course No: CSE3214
Course Title: Operating System Lab

Date of Submission: 8/7/24

Assignment No: 04

Submitted by:
Name: Maliha Akter Miti
Student ID: 20210104122

(i) Practical Implementation of Deadlock Detection using Resource Allocation Graph using C/C++

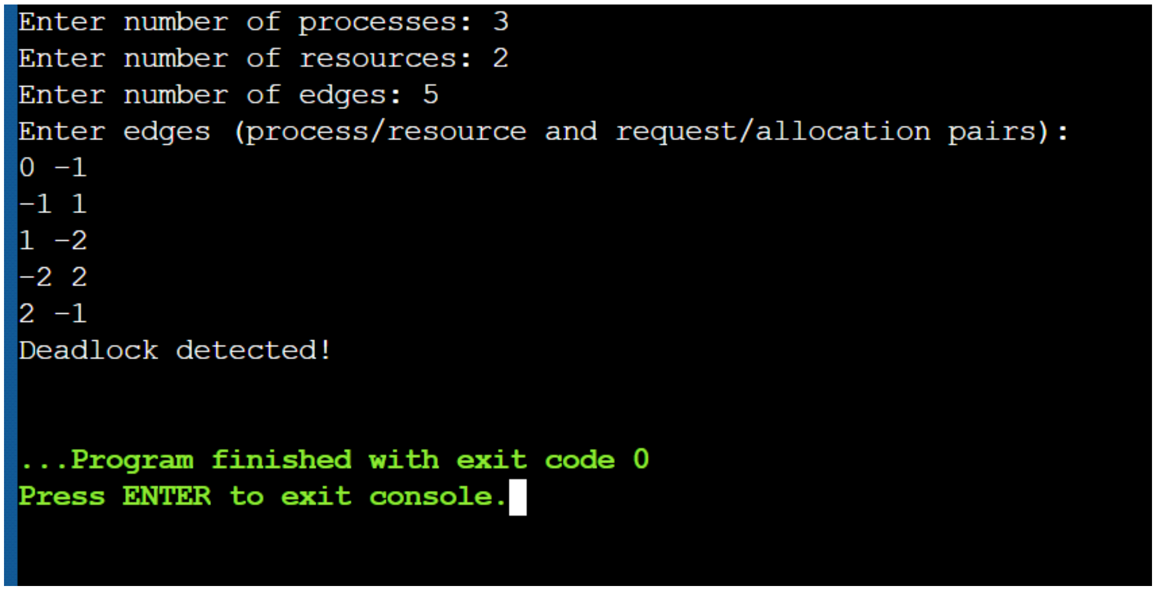
Code:

```
#include <iostream>

#include <vector>

#include <unordered_map>

#include <unordered_set>


// Graph representation

class Graph {

private:

  std::unordered_map<int, std::vector<int>> adjList;


  bool detectCycleDFS(int node, std::unordered_set<int>& visited, std::unordered_set<int>& recStack) {

    if (recStack.find(node) != recStack.end()) return true;

    if (visited.find(node) != visited.end()) return false;


    visited.insert(node);

    recStack.insert(node);


    for (int neighbor : adjList[node]) {

      if (detectCycleDFS(neighbor, visited, recStack)) {

        return true;

      }

    }
```

```
        recStack.erase(node);
        return false;
    }

public:
    void addEdge(int u, int v) {
        adjList[u].push_back(v);
    }

    bool detectCycle() {
        std::unordered_set<int> visited;
        std::unordered_set<int> recStack;

        for (const auto& pair : adjList) {
            int node = pair.first;
            if (detectCycleDFS(node, visited, recStack)) {
                return true;
            }
        }

        return false;
    }
};
```

```
int main() {
    Graph g;
    int numProcesses, numResources, numEdges;

    std::cout << "Enter number of processes: ";
    std::cin >> numProcesses;
    std::cout << "Enter number of resources: ";
    std::cin >> numResources;
    std::cout << "Enter number of edges: ";
    std::cin >> numEdges;

    std::cout << "Enter edges (process/resource and request/allocation pairs):" << std::endl;
    for (int i = 0; i < numEdges; ++i) {
        int u, v;
        std::cin >> u >> v;
        g.addEdge(u, v);
    }

    if (g.detectCycle()) {
        std::cout << "Deadlock detected!" << std::endl;
    } else {
        std::cout << "No deadlock detected." << std::endl;
    }

    return 0;
}
```

Screenshot:



```
Enter number of processes: 3
Enter number of resources: 2
Enter number of edges: 5
Enter edges (process/resource and request/allocation pairs):
0 -1
-1 1
1 -2
-2 2
2 -1
Deadlock detected!


...Program finished with exit code 0
Press ENTER to exit console.
```