Maarij Khan

11/22/2024

Example of program:

BankersAlgorithm.cpp

```
#include <iostream>

#include <fstream>

#include <vector>


using namespace std;



void computeRequired(vector<vector<int>>& required, vector<vector<int>>& maximum,
vector<vector<int>>& allocated, int numProcesses, int numResources) {

   for (int i = 0; i < numProcesses; i++) {

      for (int j = 0; j < numResources; j++) {

         required[i][j] = maximum[i][j] - allocated[i][j];

      }

   }

}



bool checkSafety(vector<vector<int>>& allocated, vector<vector<int>>& required, vector<int>&
available, int numProcesses, int numResources) {

   vector<bool> finished(numProcesses, false);  // Tracks which processes have completed

   vector<int> safeOrder(numProcesses);       // Stores the safe sequence

   vector<int> work = available;              // Copy of available resources

   int count = 0;


   while (count < numProcesses) {
```

```
    bool foundProcess = false;
    for (int i = 0; i < numProcesses; i++) {
        if (!finished[i]) {
            bool canProceed = true;
            for (int j = 0; j < numResources; j++) {
                if (required[i][j] > work[j]) {
                    canProceed = false;
                    break;
                }
            }
            if (canProceed) {
                for (int j = 0; j < numResources; j++) {
                    work[j] += allocated[i][j];
                }
                safeOrder[count++] = i;
                finished[i] = true;
                foundProcess = true;
            }
        }
    }
    if (!foundProcess) {
        cout << "The system is not in a safe state.\n";
        return false;
    }
}

cout << "The system is in a safe state.\nSafe sequence is: ";
for (int i = 0; i < numProcesses; i++) {
    cout << "P" << safeOrder[i];
```

```
        if (i != numProcesses - 1) cout << " -> ";
    }
    cout << endl;
    return true;
}

int main() {
    ifstream inputFile("C:\\Users\\13303\\.ssh\\input.txt");
    if (!inputFile) {
        cerr << "Error: Could not open the input file.\n";
        return 1;
    }

    int numProcesses = 5, numResources = 3;

    vector<vector<int>> allocated(numProcesses, vector<int>(numResources));
    vector<vector<int>> maximum(numProcesses, vector<int>(numResources));
    vector<vector<int>> required(numProcesses, vector<int>(numResources));
    vector<int> available(numResources);

    // Reading data from input.txt
    for (int i = 0; i < numProcesses; i++) {
        for (int j = 0; j < numResources; j++) {
            inputFile >> allocated[i][j];
        }
    }

    for (int i = 0; i < numProcesses; i++) {
        for (int j = 0; j < numResources; j++) {
```

```
            inputFile >> maximum[i][j];
        }
    }

    for (int i = 0; i < numResources; i++) {
        inputFile >> available[i];
    }

    // Compute the required resources matrix
    computeRequired(required, maximum, allocated, numProcesses, numResources);

    // Check if the system is in a safe state
    checkSafety(allocated, required, available, numProcesses, numResources);

    return 0;
}
```

Example of Input.txt 1–

```
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2

7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
```

1 0 0

Output: The system is not in a safe state.

Example of Input.txt 2–

0 1 0

2 0 0

3 0 2

2 1 1

0 0 2

7 5 3

3 2 2

9 0 2

2 2 2

4 3 3

3 3 2

Output: The system is in a safe state.

Safe sequence is: P1 -> P3 -> P4 -> P0 -> P2