CSC 345: Home Work 4
The Bankers safety algorithm

Table of Contents

Objective:

The Bankers safety algorithm is a keystone algorithm in both Deadlock avoidance, and Deadlock detection and recovery. In this homework your objective is to implement the Banker's algorithm as discussed in class using the following pseudo-code.

The Bankers Safety Algorithm:

Initialization

- Assume there are 'n' processes and 'm' resources.
- Data structures:
  - $Available_m$: Available [i] i.e., a vector / one dimensional array
  - Max_{nxm}: Max [i, j] i.e., a collection of vectors / a two dimensional matrix
  - $Allocation_{nxm}$: Allocation [i, j], i.e., a collection of vectors / a two dimensional matrix
  - $Need_{nxm}$: Need [i, j], i.e., a collection of vectors / a two dimensional matrix

NOTE: $Need = Max - Allocation.$

Assume a process $P_i \rightarrow Request_i$

```
Algorithm: Bankers algorithm
    1. If $Request_i \leq Need_i$ then go to step 2,
                   else error "Exceeds Needs"
    2. If $Request_i \leq Available$ then go to step 3
                   else wait
    3. $Perform\ all\ of\ the\ updates \begin{cases} Available = Available - Request_i \\ Allocation_i = Allocation_i + Request_i \\ Need_i = Need_i - Request_i \end{cases}$
    4. Check if this new state is safe i.e., if there exists a safe sequence.
       i.e., invoke the Safety Algorithm
```

CSC 345: Home Work 4
The Bankers safety algorithm

```
Algorithm: Safety Algorithm
    1  $Work = Available$, and $Finish_n = false$ (Finish is a Boolean vector)
    2  Find an $i$ such that $Finish[i] = False$, and $Need[i] \leq Work$
           If no such $i$, go to step 4
    3  $Work = Work + Allocation_i$ , $Finish[i] = true$, and go to step 2.
    4  If $Finish[i] = true$ for all $i$ then system is safe.
```

Your Task

Implement the above Bankers Safety algorithm:

1. Your implementation to check for safety, and
2. Generate the safe sequence, for any number of processes and resources
3. Generate the safe sequences for the following inputs (a) & (b) respectively

Input

Initialize the system

n= number of processes = 5 , i.e., p0, p1, p2, p3, and p4
m= number of resources = 3, i.e., A, B, and C.
Resource instances: A=10, B=5, C=7.

Initial Allocations:

	A	B	C
p0	0	1	0
p1	2	0	0
p2	3	0	2
p3	2	1	1
p4	0	0	2

Maximum Requirements for each process:

	A	B	C
p0	7	5	3
p1	3	2	2
p2	9	0	2
p3	2	2	2
p4	4	3	3

Generate Safety sequence

Generated by the request placed by process p1
Request =<1 0 2>

Report:

Your report should consist of the following:

□ Your code
□ Your make file (very important)
□ Outputs of your code, for the input specification (above)