Write a C program to simulate bankers’ algorithm for the purpose of deadlock detection.

**Program:**

#include <stdio.h>

#include <stdbool.h>

int main()

{

int n, m, i, j, k;

printf("Enter number of processes and resources:\n");

scanf("%d %d", &n, &m);

int alloc[n][m], request[n][m], avail[m];

bool finish[n];

int safe\_sequence[n];

int count =0;

printf("Enter allocation matrix:\n");

for (i = 0; i < n; i++)

for (j = 0; j < m; j++)

scanf("%d", &alloc[i][j]);

printf("Enter request matrix:\n");

for (i = 0; i < n; i++)

for (j = 0; j < m; j++)

scanf("%d", &request[i][j]);

printf("Enter available matrix:\n");

for (i = 0; i < m; i++)

scanf("%d", &avail[i]);

for (i = 0; i < n; i++)

{

bool is\_zero = true;

for (j = 0; j < m; j++)

{

if (alloc[i][j] != 0)

{

is\_zero = false;

break;

}

}

finish[i] = is\_zero;

}

bool changed;

do

{

changed = false;

for (i = 0; i < n; i++)

{

if (!finish[i])

{

bool can\_finish = true;

for (j = 0; j < m; j++)

{

if (request[i][j] > avail[j])

{

can\_finish = false;

break;

}

}

if (can\_finish)

{

for (k = 0; k < m; k++)

avail[k] += alloc[i][k];

finish[i] = true;

changed = true;

printf("Process %d can finish.\n", i);

safe\_sequence[count++] = i;

}

else{

printf("Process %d causes deadlock.\n");

break;

}

}

}

} while (changed);

bool deadlock = false;

for (i = 0; i < n; i++)

{

if (!finish[i])

{

deadlock = true;

break;

}

}

if (deadlock)

printf("System is in a deadlock state.\n");

else

{

printf("System is not in a deadlock state.\n");

printf("Safe sequence is: ");

for (int i = 0; i < n; i++)

{

printf("P%d", safe\_sequence[i]);

if (i != n - 1)

{

printf(" -> ");

}

}

}

return 0;

}

**Output:**



