// Program for Deadlock detection algorithm

#include <stdio.h>

#include <stdbool.h>

#define MAX\_PROCESSES 10

#define MAX\_RESOURCES 10

int processes, resources;

int available[MAX\_RESOURCES];

int allocation[MAX\_PROCESSES][MAX\_RESOURCES];

int request[MAX\_PROCESSES][MAX\_RESOURCES];

void deadlockDetection() {

bool finish[MAX\_PROCESSES] = {false};

int work[MAX\_RESOURCES];

// Initialize work as a copy of available resources

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

work[i] = available[i];

}

bool deadlock = false;

int deadlockedProcesses[MAX\_PROCESSES];

int deadlockedCount = 0;

for (int count = 0; count < processes; count++) {

bool found = false;

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

if (!finish[i]) {

bool canProceed = true;

// Check if the process's request can be satisfied

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

if (request[i][j] > work[j]) {

canProceed = false;

break;

}

}

// If the request can be satisfied, allocate resources temporarily

if (canProceed) {

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

work[j] += allocation[i][j];

}

finish[i] = true;

found = true;

}

}

}

// If no process could proceed in this round, break out

if (!found) {

break;

}

}

// Check for processes still marked as unfinished

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

if (!finish[i]) {

deadlockedProcesses[deadlockedCount++] = i;

deadlock = true;

}

}

if (deadlock) {

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

printf("Deadlocked processes: ");

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

printf("P%d ", deadlockedProcesses[i]);

}

printf("\n");

} else {

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

}

}

int main() {

printf("Enter the number of processes: ");

scanf("%d", &processes);

printf("Enter the number of resources: ");

scanf("%d", &resources);

printf("Enter the available resources:\n");

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

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

}

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

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

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

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

}

}

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

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

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

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

}

}

deadlockDetection();

return 0;

}