**LAB-7**

Q. Write a program to implement a Banker’s algorithm.

**Code:**

#include <stdio.h>

int main() {

int num\_processes, num\_resources;

printf("Enter number of processes: ");

scanf("%d", &num\_processes);

printf("Enter number of resources: ");

scanf("%d", &num\_resources);

int available[num\_resources];

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

for (int i = 0; i < num\_resources; i++) {

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

}

int max[num\_processes][num\_resources];

int allocation[num\_processes][num\_resources];

int need[num\_processes][num\_resources];

printf("Enter the maximum resources for each process:\n");

for (int i = 0; i < num\_processes; i++) {

printf("Process %d:\n", i);

for (int j = 0; j < num\_resources; j++) {

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

}

}

printf("Enter the allocated resources for each process:\n");

for (int i = 0; i < num\_processes; i++) {

printf("Process %d:\n", i);

for (int j = 0; j < num\_resources; j++) {

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

need[i][j] = max[i][j] - allocation[i][j];

}

}

int work[num\_resources];

for (int i = 0; i < num\_resources; i++) {

work[i] = available[i];

}

int finish[num\_processes];

for (int i = 0; i < num\_processes; i++) {

finish[i] = 0;

}

int safe\_sequence[num\_processes];

int num\_safe = 0;

while (num\_safe < num\_processes) {

int found = 0;

for (int i = 0; i < num\_processes; i++) {

if (!finish[i]) {

int j;

for (j = 0; j < num\_resources; j++) {

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

break;

}

}

if (j == num\_resources) {

found = 1;

for (int k = 0; k < num\_resources; k++) {

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

}

safe\_sequence[num\_safe++] = i;

finish[i] = 1;

}

}

}

if (!found) {

printf("Unsafe state!\n");

return 1;

}

}

printf("Safe sequence:");

for (int i = 0; i < num\_processes; i++) {

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

}

printf("\n");

return 0;

}

**Output:**



