Week 6

#include <stdio.h>

#include <stdbool.h>

#define MAX\_P 10

#define MAX\_R 10

int main() {

int n, m;

int alloc[MAX\_P][MAX\_R], max[MAX\_P][MAX\_R], need[MAX\_P][MAX\_R];

int avail[MAX\_R];

printf("Enter number of processes -- ");

scanf("%d", &n);

printf("Enter number of resources -- ");

scanf("%d", &m);

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

printf("Enter details for P%d\n", i);

printf("Enter allocation -- ");

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

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

printf("Enter Max -- ");

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

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

}

printf("Enter Available Resources -- ");

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

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

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

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

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

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

int work[MAX\_R];

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

work[i] = avail[i];

int safeSeq[MAX\_P], count = 0;

while (count < n) {

bool found = false;

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

if (!finish[i]) {

bool canFinish = true;

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

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

canFinish = false;

break;

}

}

if (canFinish) {

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

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

safeSeq[count++] = i;

finish[i] = true;

found = true;

printf("P%d is visited( ", i);

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

printf("%d ", work[j]);

printf(")\n");

}

}

}

if (!found) break;

}

bool safe = true;

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

if (!finish[i]) {

safe = false;

break;

}

if (safe) {

printf("SYSTEM IS IN SAFE STATE\n");

printf("The Safe Sequence is -- (");

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

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

if (i != n - 1) printf(" ");

}

printf(")\n");

} else {

printf("SYSTEM IS IN UNSAFE STATE\n");

}

printf("\nProcess\tAllocation\tMax\t\tNeed\n");

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

printf("P%d\t", i);

for (int j = 0; j < m; j++) printf("%d ", alloc[i][j]);

printf("\t\t");

for (int j = 0; j < m; j++) printf("%d ", max[i][j]);

printf("\t\t");

for (int j = 0; j < m; j++) printf("%d ", need[i][j]);

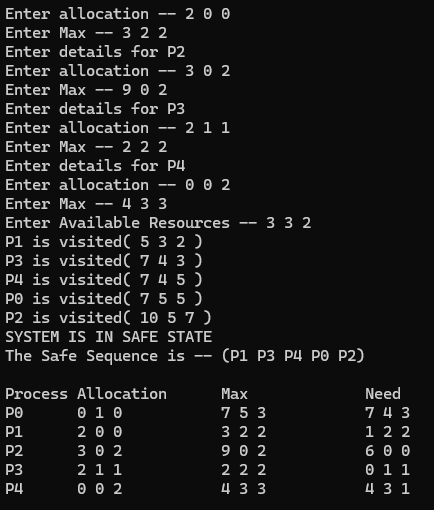
printf("\n");

}

return 0;

}

Safe state



Unsafe state

