// Banker's Algorithm

#include <stdio.h>

int main()

{

// P0, P1, P2, P3, P4 are the Process names here

int n, m, i, j, k,ys,padd,fl;

int alloc[5][3],add[10];

int max[5][3];

int avail[3];

printf("Enter no. of processes");

scanf("%d",&n);

printf("Enter no. of resources");

scanf("%d",&m);

printf("Enter alloc");

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

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

printf("Enter %d row %d ele",(i+1),(j+1));

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

}

}

printf("Enter max");

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

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

printf("Enter %d row %d ele",(i+1),(j+1));

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

}

}

printf("Enter avail");

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

printf("Enter %d ele",(i+1));

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

}

printf("Do you want additional resource? ");

scanf("%d",&ys);

if(ys==1){

printf("Enter process");

scanf("%d",&padd);

printf("Enter add requirement");

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

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

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

if(add[i]>avail[i]){

printf("Requirement more than avail");

return 0;

}

fl=0;

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

if((alloc[padd][i]+add[i])>max[padd][i]){

printf("asking for more resource than what can be granted");

fl=1;break;

}

}

if(fl==0){

printf("Requirement can be satisfied");

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

alloc[padd][i]+=add[i];

avail[i]-=add[i];

}

}

}

}

int f[n], ans[n], ind = 0;

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

{

f[k] = 0;

}

int need[n][m];

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

{

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

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

}

int y = 0;

for (k = 0; k < 5; k++)

{

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

{

if (f[i] == 0)

{

int flag = 0;

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

{

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

{

flag = 1;

break;

}

}

if (flag == 0)

{

ans[ind++] = i;

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

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

f[i] = 1;

break;

}

}

}

}

int flag = 1;

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

{

if (f[i] == 0)

{

flag = 0;

printf("The following system is not safe");

break;

}

}

if (flag == 1)

{

printf("Following is the SAFE Sequence\n");

for (i = 0; i < n - 1; i++)

printf(" P%d ->", ans[i]);

printf(" P%d", ans[n - 1]);

}

return (0);

}

