#include <stdio.h>

int main()

{

int numP, numR;

int y = 0;

printf("enter number of processes\n");

scanf("%d", &numP);

printf("\nenter number of resources\n");

scanf("%d", &numR);

int alot[numP][numR];

int max[numP][numR];

int need[numP][numR];

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

{

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

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

{

printf("\n enter no. of alloc resource %c : ", (65 + j));

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

}

printf("\n");

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

{

printf("\n enter no. of max resource %c : ", (65 + j));

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

}

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

{

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

}

}

int visited[numP];

int avai[numR];

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

{

printf("\n enter no. of avai resource %c :", (65 + i));

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

}

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

{

visited[i] = 0;

}

// printing table

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

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

{

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

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

{

printf("%d ", alot[i][j]);

}

printf("\t\t");

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

{

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

}

printf("\t\t");

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

{

printf("%d ", need[i][j]);

}

printf("\n");

}

printf("\navailable\n");

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

{

printf("%c : ", (65 + i));

printf("%d ", avai[i]);

}

printf("\ndo u want to enter request (1/0) enter integer\n");

scanf("%d", &y);

if (y == 1)

{

int p, req, flag = 1; // flag will set the premission

int newAvai[numR];

printf("enter process number :");

scanf("%d", &p);

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

{

printf("\n enter no. of request resource %c :", (65 + i));

scanf("%d", &req);

// check request is less then need & avail

if (avai[i] < req && need[p][i] < req)

{

flag = 0;

break;

}

else

{

newAvai[i] = avai[i] - req;

alot[p][i] = alot[p][i] + req;

}

}

if (!flag)

{

printf("\naccess not granted");

}

else

{

int safe[numP], safeseq = 0;

// this section will check that do any process recived resource in 1 iteration , if no one gets resource means its deadlock

while (safeseq < numP && flag != 0)

{

flag = 0; // means no process is given resource

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

{

if (need[i][0] <= newAvai[0] && visited[i] != 1)

{

int f = 1;

for (int j = 1; j < numR; j++)

{

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

{

f = 0;

break;

}

}

if (f)

{

flag = 1; // a process is given resource

visited[i] = 1;

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

{

newAvai[j] = newAvai[j] + alot[i][j];

}

safe[safeseq] = i;

safeseq++;

}

}

}

}

if (flag)

{

printf("\naccess granted");

printf("\nsafe sequence :\n");

for (int a = 0; a < numP; a++)

{

printf("%d->", safe[a]);

}

printf("\n==============finish condition==================\n");

printf("P%d finish[%d] safe\_index = %d and safe[%d] = %d", safe[safeseq - 1], safe[safeseq - 1], visited[safe[safeseq - 1]], safeseq - 1, safe[safeseq - 1]);

printf("\n========================================\n");

printf("\ntotal resources\n");

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

{

printf("%c : ", (65 + i));

printf("%d ", newAvai[i]);

}

}

else

{

printf("\naccess not granted");

}

}

}

return 0;

}