// Task-1:

#include <stdio.h>

/\*Sample Input:

Number of processes:

5

Number of recources:

4

Allocation Matrix:

0 1 0 3

2 0 0 0

3 0 2 0

2 1 1 5

0 0 2 2

Maximum Matrix:

6 4 3 4

3 2 2 1

9 1 2 6

2 2 2 8

4 3 3 7

Available Matrix:

10 5 7 11

\*/

int maxArray[100][100];

int alloc[100][100];

int need[100][100];

int avail[100][100];

int n, m;

void user\_data()

{

int x, y;

printf("Number of processes:\n");

scanf("%d", &n);

printf("Number of recources:\n");

scanf("%d", &m);

printf("Allocation Matrix:\n");

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

{

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

{

scanf("%d", &alloc[x][y]);

}

}

printf("Maximum Matrix:\n");

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

{

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

{

scanf("%d", &maxArray[x][y]);

}

}

printf("Available Matrix:\n");

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

{

scanf("%d", &avail[0][x]);

}

}

void calculation()

{

int finish[100], temp, need[100][100], flag = 1, k, count = 0;

int dead[100];

int safe[100];

int x, y;

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

{

finish[x] = 0;

dead[x] = 0;

safe[x] = 0;

}

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

{

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

{

need[x][y] = maxArray[x][y] - alloc[x][y];

}

}

while (flag)

{

flag = 0;

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

{

int c = 0;

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

{

if (finish[x] == 0 && need[x][y] <= avail[0][y])

{

c++;

if (c == m)

{

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

{

avail[0][k] = avail[0][k] + alloc[x][k];

finish[x] = 1;

flag = 1;

}

if (finish[x] == 1)

{

x = n;

}

}

}

}

}

}

y = 0;

flag = 0;

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

{

if (finish[x] == 0)

{

dead[y] = x;

y++;

flag = 1;

}

}

if (flag == 1)

{

printf("DEADLOCK AHEAD!\n");

}

else

{

printf("SAFE HERE!\n");

}

}

int main()

{

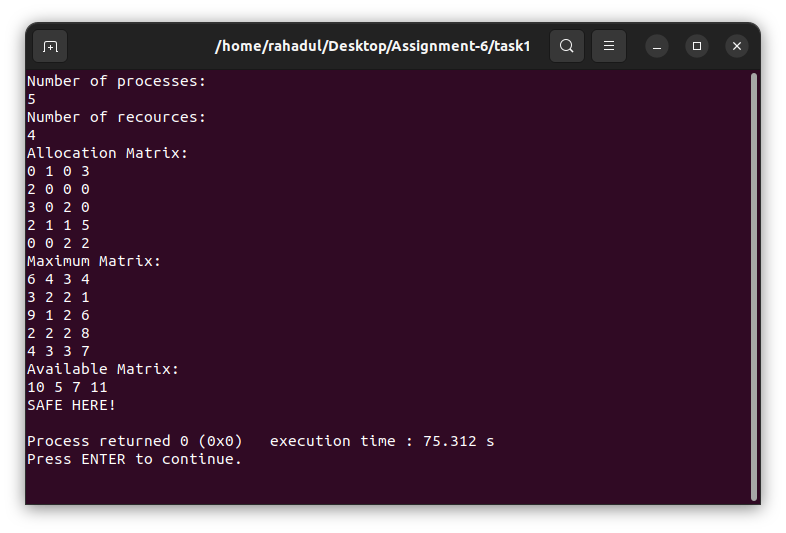
int x, y;

user\_data();

calculation();

return 0;

}



//Task-2:

#include <stdio.h>

int main()

{

int x, y, z, m, n;

n = 6;

m = 4;

char sequence[1000];

int alloc[6][4] =

{

{0, 1, 0, 3},

{2, 0, 0, 3},

{3, 0, 2, 0},

{2, 1, 1, 5},

{0, 0, 2, 2},

{1, 2, 3, 1}

};

int max[6][4] =

{

{6, 4, 3, 4},

{3, 2, 2, 4},

{9, 1, 2, 6},

{2, 2, 2, 8},

{4, 3, 3, 7},

{6, 2, 6, 5}

};

int temp[n], result[n], counter = 0;

int avail[4] = {2, 2, 2, 1};

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

{

temp[z] = 0;

}

int need[n][m];

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

{

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

{

need[x][y] = max[x][y] - alloc[x][y];

}

}

int a = 0;

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

{

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

{

if (temp[x] == 0)

{

int flag = 0;

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

{

if (need[x][a] > avail[a])

{

flag = 1;

break;

}

}

if (flag == 0)

{

result[counter] = x;

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

{

avail[a] += alloc[x][a];

}

temp[x] = 1;

counter++;

}

}

}

}

int flag = 1;

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

{

if (temp[x] == 0)

{

flag = 0;

printf("DEADLOCK AHEAD!\n");

break;

}

}

if (flag == 1)

{

printf("Safe Sequence:\n");

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

{

if (x==n-1)

{

printf("P%d", result[x]+1);

}

else

printf("P%d --> ", result[x]+1);

}

}

return 0;

printf("\n");

}

