11 YASHRAJ DEEPAK DEVRAT

#include<stdio.h>

int max[10][10],allocation[10][10],need[10][10];

int avail[10];

int np, nr;

void readmatrix(int matrix[10][10])

{

int i,j;

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

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

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

}

void display\_matrix(int matrix[10][10])

{

int i,j;

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

{

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

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

{

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

}

}

}

void calculate\_need()

{

int i,j;

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

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

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

}

void banker()

{

int i,j,k=0,flag;

int finish[10],safe\_seq[10];

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

{

finish[i]=0;//Declare as all processes are incomplete

}

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

{ flag=0;

if(finish[i]==0)//Execute incomplete processes

{

for(j=0;j<nr;j++)//check a need of each process

{

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

{

flag=1;//Break a loop as need is greater than avail and go to next process

break;

}

}

if(flag==0)//Need is lesser than avail so complete process

{

finish[i]=1;

safe\_seq[k]=i;

k++;

//Add allocated resources of finished process in available resources.

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

avail[j]+=allocation[i][j];

//start checking from first process again.

i=-1;

}

}

}

flag=0;//Check if all processes are completed

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

{

if(finish[i]==0)

{

printf("\nThe system is in deadlock");

flag=1;

break;

}

}

if(flag==0)

{

printf("\n The system is in safe state! \n Safe sequence is ==>");

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

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

}

}

int main()

{

int j;

//read input

{

printf("\nEnter number of processes");

scanf("%d",&np);

printf("\nEnter number of resources");

scanf("%d",&nr);

printf("\n Enter initial allocation matrix:");

readmatrix(allocation);

printf("\n Enter Max requirement matrix:");

readmatrix(max);

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

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

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

}

//Display entered data

{

printf("\n \*\*\*\*\*\*Entered Data is \*\*\*\*\*\*\*\*\n\n");

printf("\n Initial allocation:\n");

display\_matrix(allocation);

printf("\n\n\n Maximum Requirement\n");

display\_matrix(max);

printf("\n Available Resources\n");

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

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

}

//Calculate and display need

{

calculate\_need();

printf("\n\n\n Need is \n");

display\_matrix(need);

}

//Execute proceeses using Bankers Algorithem

banker();

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

return 0;

}





