PROGRAm

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

void \*func1();

void \*func2();

pthread\_mutex\_t lk;

int Max\_resource[20][20];

int need\_resource[20][20];

int allocated\_resource[20][20];

int available\_resource[20];

int complete\_process[20];

int safe\_sequence[20];

int prc, res, i, j, process, count=0;

int main()

{

pthread\_t p1,p2;

pthread\_create(&p1,NULL,\*func1,NULL);

pthread\_create(&p1,NULL,\*func2,NULL);

pthread\_join(p1,NULL);

pthread\_join(p2,NULL);

}

void \*func2()

{

pthread\_mutex\_lock(&lk);

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

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

{

need\_resource[i][j] = Max\_resource[i][j] - allocated\_resource[i][j];

}

do

{

printf("\n Maximum need resource : Allocated resource:\t Need resources\n");

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

{

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

{

printf("%d ", Max\_resource[i][j]);

}

printf("\t\t\t\t");

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

{

printf("%d ", allocated\_resource[i][j]);

}

printf("\t\t\t");

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

{

printf("%d ", Max\_resource[i][j] - allocated\_resource[i][j]);

}

printf("\n");

}

process = -1;

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

{

if(complete\_process[i] == 0)

{

process = i ;

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

{

if(available\_resource[j] < need\_resource[i][j])

{

process = -1;

break;

}

}

}

if(process != -1)

break;

}

if(process != -1)

{

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

printf("Process %d executed", process + 1);

safe\_sequence[count] = process + 1;

count++;

printf("\nNew available resource : ");

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

{

available\_resource[j] += allocated\_resource[process][j];

allocated\_resource[process][j] = 0;

Max\_resource[process][j] = 0;

complete\_process[process] = 1;

printf(" %d ",available\_resource[j]);

}

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

}

}

while(count != prc && process != -1);

if(count == prc)

{

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

printf("\nThe system is in a safe state\n");

printf("Safe Sequence for executed process : < ");

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

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

printf(">\n");

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

}

else

{

printf("\nThe system is in an unsafe state");

}

pthread\_mutex\_unlock(&lk);

}

void \*func1()

{

pthread\_mutex\_lock(&lk);

printf("Enter how many process you want to execute : ");

scanf("%d", &prc);

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

{

complete\_process[i] = 0;

}

printf("\n\nEnter how many resouces are available : ");

scanf("%d", &res);

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*Enter how many maximum resouces need for each process \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \n");

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

{

printf("\nEnter the resouce for process %d : ", i + 1);

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

{

scanf("%d",&Max\_resource[i][j]);

}

}

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*Enter the how much resources already allocated for each process \*\*\*\*\*\*\*\*\*\*\*\*\*\* \n");

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

{

printf("\nEnter resouces for process %d : ",i + 1);

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

{

scanf("%d", &allocated\_resource[i][j]);

}

}

printf("\n\nEnter the total Available Resources to execute process : ");

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

{

scanf("%d", &available\_resource[i]);

}

pthread\_mutex\_unlock(&lk);

}