***CSE321 Lab Assignment 06***

Name: Mehnaz Ara Fazal

ID:20301295

Section:01

Email: [mehnaz.ara.fazal@g.bracu.ac.bd](mailto:mehnaz.ara.fazal@g.bracu.ac.bd)

Task-1

#include <stdio.h>

int main(){

int processes, resources;

printf("Provide no. of processes\n");

scanf("%d",&processes);

printf("Provide no. of resources\n");

scanf("%d",&resources);

int alloc[processes][resources];

int max[processes][resources];

int total[resources];

int avail[resources];

//Updating allocated array

for(int i = 0;i<processes;i++){

printf("P%d process's allocated resources\n",i);

for(int j = 0;j<resources;j++){

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

}

}

//Updating max array

for(int i = 0;i<processes;i++){

printf("P%d process's max resources\n",i);

for(int j = 0;j<resources;j++){

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

}

}

printf("total resources\n");

for(int i = 0;i<resources;i++){

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

}

//Updating avail array

for(int i=0;i<resources;i++){

int count = 0;

for(int j=0;j<processes;j++){

count += alloc[j][i];

}

avail[i] = total[i] - count;

}

//Updating need array

int need[processes][resources];

for(int i = 0;i<processes;i++){

for(int j = 0;j<resources;j++){

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

}

}

int sum\_alloc[processes];

for(int i =0;i<processes;i++){

sum\_alloc[i] = 1;

}

int process\_count = 0; //deadlock finder

for(int i=0;i<processes;i++){

for(int k=0;k<processes;k++){

int count = 0;

if (sum\_alloc[k] == 1){

for(int j=0;j<resources;j++){

if (need[k][j] <= avail[j]){

count ++;

}

}

}

if(count == resources){

process\_count ++;

sum\_alloc[k] = 0;

for(int n=0;n<resources;n++){

avail[n] += alloc[k][n];

}

}

}

}

if(process\_count == processes){

printf("SAFE HERE!");

}

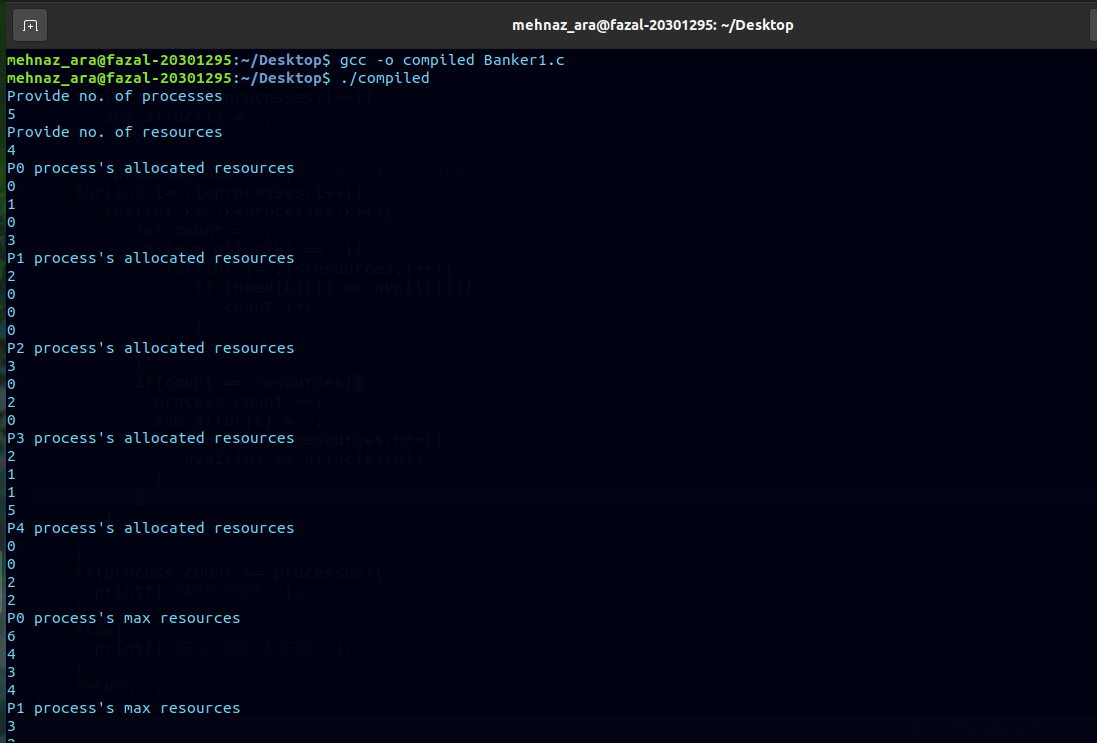
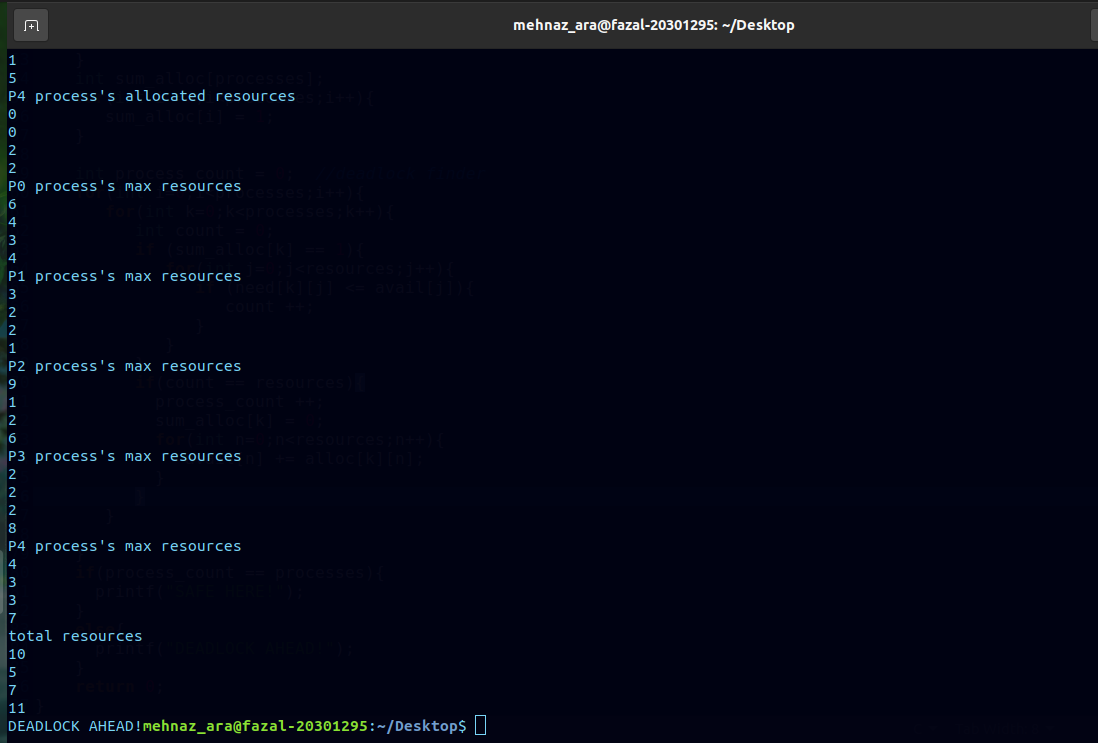
else{

printf("DEADLOCK AHEAD!");

}

return 0;

}

Task-2

#include <stdio.h>

int main(){

int processes, resources;

printf("Provide no. of processes\n");

scanf("%d",&processes);

printf("Provide no. of resources\n");

scanf("%d",&resources);

int alloc[processes][resources];

int max[processes][resources];

int total[resources];

int avail[resources];

//Updating allocated array

for(int i = 0;i<processes;i++){

printf("P%d process's allocated resources\n",i);

for(int j = 0;j<resources;j++){

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

}

}

//Updating max array

for(int i = 0;i<processes;i++){

printf("P%d process's max resources\n",i);

for(int j = 0;j<resources;j++){

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

}

}

printf("total resources\n");

for(int i = 0;i<resources;i++){

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

}

//Updating avail array

for(int i=0;i<resources;i++){

int count = 0;

for(int j=0;j<processes;j++){

count += alloc[j][i];

}

avail[i] = total[i] - count;

}

//Updating need array

int need[processes][resources];

for(int i = 0;i<processes;i++){

for(int j = 0;j<resources;j++){

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

}

}

int sum\_alloc[processes];

for(int i =0;i<processes;i++){

sum\_alloc[i] = 1;

}

int safe\_seq[processes];

int safe\_seq\_ptr = 0;

int process\_count = 0; //deadlock finder

for(int i=0;i<processes;i++){

for(int k=0;k<processes;k++){

int count = 0;

if (sum\_alloc[k] == 1){

for(int j=0;j<resources;j++){

if (need[k][j] <= avail[j]){

count ++;

}

}

}

if(count == resources){

process\_count ++;

sum\_alloc[k] = 0;

safe\_seq[safe\_seq\_ptr] = k;

safe\_seq\_ptr ++;

for(int n=0;n<resources;n++){

avail[n] += alloc[k][n];

}

}

}

}

if(process\_count == processes){

safe\_seq\_ptr = safe\_seq\_ptr - 1;

printf("Safe Sequence:\n");

for(int i=0;i<processes;i++){

if(i != safe\_seq\_ptr){

printf("P%d-->",safe\_seq[i]+1);

}

else{

printf("P%d\n",safe\_seq[i]+1);

}

}

}

else{

printf("DEADLOCK AHEAD!");

}

return 0;

}

