

**“lab 12 &13”**

**COURSE :**

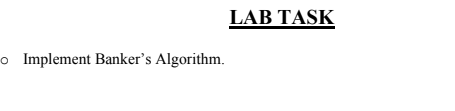
**OPERATING SYSTEM SUBMITTED TO :**

**Sir shehzad**

**SUBMITTED BY :**

**Rabia Batool (2022-BSE-064) SECTION :**

**B**



Code:

#include <stdio.h>

#include <stdbool.h>

struct process\_info

{

int max[10];

int allocated[10];

int need[10];

};

int no\_of\_process,no\_of\_resources;

void input(struct process\_info process[no\_of\_process],int available[no\_of\_resources])

{

for(int i=0;i<no\_of\_process;i++)

{

printf("Enter process[%d] info\n",i);

printf("Enter Maximum Need: ");

for(int j=0;j<no\_of\_resources;j++)

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

printf("Enter No. of Allocated Resources for this process: ");

for(int j=0;j<no\_of\_resources;j++)

{

scanf("%d",&process[i].allocated[j]);

process[i].need[j]= process[i].max[j] - process[i].allocated[j];

}

}

printf("Enter Available Resources: ");

for(int i=0;i<no\_of\_resources;i++)

{

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

}

}

void showTheInfo(struct process\_info process[no\_of\_process])

{

printf("\nPID\tMaximum\t\tAllocated\tNeed\n");

for(int i=0;i<no\_of\_process;i++)

{

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

for(int j=0;j<no\_of\_resources;j++)

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

printf("\t\t");

for(int j=0;j<no\_of\_resources;j++)

printf("%d ",process[i].allocated[j]);

printf("\t\t");

for(int j=0;j<no\_of\_resources;j++)

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

printf("\n");

}

}

bool applySafetyAlgo(struct process\_info process[no\_of\_process],int available[no\_of\_resources],int safeSequence[no\_of\_process])

{

bool finish[no\_of\_process];

int work[no\_of\_resources];

for(int i=0;i<no\_of\_resources;i++)

{

work[i]=available[i];

}

for(int i=0;i<no\_of\_process;i++)

finish[i]=false;

bool proceed=true;

int k=0;

while(proceed)

{

proceed=false;

for(int i=0;i<no\_of\_process;i++)

{

bool flag=true;

if(finish[i]==false)

{

for(int j=0;j<no\_of\_resources;j++)

{

if(process[i].need[j] <= work[j])

{

continue;

}

else

{

flag=false;

break;

}

}

if(flag==false)

continue;

for(int j=0;j<no\_of\_resources;j++)

work[j]=work[j]+ process[i].allocated[j];

finish[i]=true;

safeSequence[k++]=i;

proceed=true;

}

}

}

int i;

for( i=0;i<no\_of\_process&&finish[i]==true;i++)

{

continue;

}

if(i==no\_of\_process)

return true;

else

return false;

}

bool isSafeState(struct process\_info process[no\_of\_process],int available[no\_of\_resources],int safeSequence[no\_of\_process])

{

if(applySafetyAlgo(process,available,safeSequence)==true)

return true;

return false;

}

int main()

{

printf("Enter No of Process\n");

scanf("%d",&no\_of\_process);

printf("Enter No of Resource Instances in system\n");

scanf("%d",&no\_of\_resources);

int available[no\_of\_resources];

int safeSequence[no\_of\_process];

struct process\_info process[no\_of\_process];

printf("\*\*\*\*\*\*\*\*\*\*\*Enter details of processes\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

input(process,available);

showTheInfo(process);

if(isSafeState(process,available,safeSequence))

{

printf("\nSystem is in SAFE State\n");

printf("Safe Sequence is: ");

for(int i=0;i<no\_of\_process;i++)

printf("P[%d] ",safeSequence[i]);

printf("1");

}

else

printf("0");

return 0;

}

