OPERATING SYSTEMS SWE2007

LAB ASSESMENT

TERALA SRUJAN-18MIS7202

1.Implementation of Deadlock Prevention Using Banker's Algorithm.

```
#include<stdio.h>
int max[100][100];
int alloc[100][100];
int need[100][100];
int avail[100];
int n,r;
void input();
void show();
void cal();
int main()
{
 int i,j;
printf(" --------TERALA SRUJAN 18MIS7202------------\n");
 printf(" ------------Banker's Algorithm------------\n");
 input();
 show();
 cal();
 getch();
 return 0;
}
void input()
{
```

```
 int i,j;
 printf("Enter the no of Processes\t");
 scanf("%d",&n);
 printf("Enter the no of resources instances\t");
 scanf("%d",&r);
 printf("Enter the Max Matrix\n");
 for(i=0;i<n;i++)
 {
  for(j=0;j<r;j++)
  {
   scanf("%d",&max[i][j]);
  }
 }
 printf("Enter the Allocation Matrix\n");
 for(i=0;i<n;i++)
 {
  for(j=0;j<r;j++)
  {
   scanf("%d",&alloc[i][j]);
  }
 }
 printf("Enter the available Resources\n");
 for(j=0;j<r;j++)
 {
  scanf("%d",&avail[j]);
 }
}
void show()
{
 int i,j;
 printf("Process\t Allocation\t Max\t Available\t");
 for(i=0;i<n;i++)
 {
```

```
  printf("\nP%d\t    ",i+1);
  for(j=0;j<r;j++)
  {
   printf("%d ",alloc[i][j]);
  }
  printf("\t");
  for(j=0;j<r;j++)
  {
   printf("%d ",max[i][j]);
  }
  printf("\t");
  if(i==0)
  {
   for(j=0;j<r;j++)
   printf("%d ",avail[j]);
  }
 }
}
void cal()
{
 int finish[100],temp,need[100][100],flag=1,k,c1=0;
 int safe[100];
 int i,j;
 for(i=0;i<n;i++)
 {
  finish[i]=0;
 }

 for(i=0;i<n;i++)
 {
  for(j=0;j<r;j++)
  {
   need[i][j]=max[i][j]-alloc[i][j];
```

```
 }
}
printf("\n");
while(flag)
{
 flag=0;
   for(i=0;i<n;i++)
 {
  int c=0;
  for(j=0;j<r;j++)
  {
   if((finish[i]==0)&&(need[i][j]<=avail[j]))
   {
       c++;
    if(c==r)
    {
     for(k=0;k<r;k++)
     {
      avail[k]+=alloc[i][j];
      finish[i]=1;
      flag=1;
          }
      printf("P%d->",i);
      if(finish[i]==1)
     {
       i=n;
      }
          }
   }
  }
   }
}
 for(i=0;i<n;i++)
```

```
 {
 if(finish[i]==1)
 {
  c1++;
 }
 else
 {
  printf("P%d->",i);
 }
}
if(c1==n)
{
 printf("\n The system is in safe state");
}
else
{
 printf("\n Process are in dead lock");
 printf("\n System is in unsafe state");
}
}
```

Output:

```
---------TERALA SRUJAN 18MIS7202----------
 ----------Banker's Algorithm----------
Enter the no of Processes       5
Enter the no of resources instances     3
Enter the Max Matrix
4
5
6
4
5
9
```

1

2

3

4

3

2

1

4

5

Enter the Allocation Matrix

4

2

3

5

1

2

3

1

5

6

3

2

4

6

7

Enter the available Resources

7

0

2

Process	Allocation	Max	Available
P1	4 2 3	4 5 6	7 0 2
P2	5 1 2	4 5 9	
P3	3 1 5	1 2 3	
P4	6 3 2	4 3 2	
P5	4 6 7	1 4 5	

P3->P2->P0->P1->P4->

The system is in safe state