EX NO: 7B	**DEADLOCK DETECTION – BANKER'S ALGORITHM**
DATE:	

AIM

To implement deadlock detection by using Banker's Algorithm.

ALGORITHM:

1. Mark each process that has a row in the Allocation matrix of all zeros.

2. Initialize a temporary vector**W** to equal the Available vector.

3. Find an index*i* such that process*i* is currently unmarked and the*i* th row of**Q** is less than or equal to **W** . That is,*Q ik … Wk*, for 1 … *k* … *m* . If no such row is found, terminate the algorithm.

4. If such a row is found, mark process*i* and add the corresponding row of the allocation matrix to **W** . That is, set*Wk = Wk* + A*ik*, for 1 … *k* … *m* . Return to step 3.

PROGRAM:

```
#include<stdio.h> static
int mark[20];
int i,j,np,nr;

int main() {
int    alloc[10][10],request[10][10],avail[10],r[10],w[10];
printf("\nEnter the no of process: "); scanf("%d",&np);
printf("\nEnter the no of resources: ");
scanf("%d",&nr); for(i=0;i<nr;i++)
{
printf("\nTotal Amount of the Resource R%d: ",i+1); scanf("%d",&r[i]);
}
printf("\nEnter the request matrix:");

for(i=0;i<np;i++) for(j=0;j<nr;j++)
scanf("%d",&request[i][j]);

printf("\nEnter the allocation matrix:");
for(i=0;i<np;i++) for(j=0;j<nr;j++)
scanf("%d",&alloc[i][j]); /*Available
Resource calculation*/ for(j=0;j<nr;j++)
{ avail[j]=r[j];
for(i=0;i<np;i++)
{
avail[j]-=alloc[i][j];
}
}
//marking processes with zero allocation for(i=0;i<np;i++)
{ int count=0;
 for(j=0;j<nr;j++)
   {
if(alloc[i][j]==0)
count++;
      else
break;
    }
 if(count==nr)  mark[i]=1;
}
```

```
// initialize W with avail

for(j=0;j<nr;j++)
   w[j]=avail[j];

//mark processes with request less than or equal to W for(i=0;i<np;i++)
{ int canbeprocessed=0;
if(mark[i]!=1) {
  for(j=0;j<nr;j++)
   {
     if(request[i][j]<=w[j])
canbeprocessed=1;       else
        {
        canbeprocessed=0;
break;
         }
    }
if(canbeprocessed)
{ mark[i]=1;

for(j=0;j<nr;j++) w[j]+=alloc[i][j];
}
}
}

//checking for unmarked processes
int deadlock=0; for(i=0;i<np;i++)
if(mark[i]!=1) deadlock=1;
if(deadlock) printf("\n Deadlock
detected"); else
printf("\n No Deadlock possible");
}
```

OUTPUT:

Enter the no of process: 4
Enter the no of resources: 5

Total Amount of the Resource R1: 2
Total Amount of the Resource R2: 1
Total Amount of the Resource R3: 1
Total Amount of the Resource R4: 2
Total Amount of the Resource R5: 1

Enter the request matrix:0 1 0 0 1
0 0 1 0 1
0 0 0 0 1
1 0 1 0 1

Enter the allocation matrix:1 0 1 1 0
1 1 0 0 0
0 0 0 1 0
0 0 0 0 0

Deadlock detected

RESULT:

Thus the banker's algorithm is implemented successfully for Deadlock detection.