5. Develop a C program to simulate Bankers Algorithm for Dead Lock Avoidance.

ALGORITHM:

1. Active:= Running U Blocked;

for k=1…r

New_ request[k]:= Requested_ resources[requesting_ process, k];

2. Simulated_ allocation:= Allocated_ resources;

for k=1…..r //Compute projected allocation state

Simulated_ allocation [requesting _process, k]:= Simulated_ allocation [requesting _process, k] + New_ request[k];

3. feasible:= true;

for k=1….r // Check whether projected allocation state is feasible

0 seconds of 0 secondsVolume 0% if

Total_ resources[k]< Simulated_ total_ alloc [k] then feasible:= false;

4. if feasible= true

then // Check whether projected allocation state is a safe allocation state

while set Active contains a process P1 such that

For all k, Total _resources[k] – Simulated_ total_ alloc[k]>= Max_ need [l ,k]-Simulated_ allocation[l, k]

Delete Pl from Active;

for k=1…..r

Simulated_ total_ alloc[k]:= Simulated_ total_ alloc[k]- Simulated_ allocation[l, k];

5. If set Active is empty

then // Projected allocation state is a safe allocation state

for k=1….r // Delete the request from pending requests

Requested_ resources [requesting_ process, k]:=0;

for k=1….r // Grant the request

Allocated_ resources [requesting_ process, k]:= Allocated_ resources[requesting_ process, k] + New_ request[k];

Total_ alloc[k]:= Total_ alloc[k] + New_ request[k];

CODE:

```
#include<stdio.h>
int main()

{
   int n,r,i,j,k,p,u=0,s=0,m;
   int block[10],run[10],active[10],newreq[10];
   int max[10][10],resalloc[10][10],resreq[10][10];
   int totalloc[10],totext[10],simalloc[10];

   printf("Enter the no of processes:");
   scanf("%d",&n);
   printf("Enter the no ofresource classes:");
   scanf("%d",&r);
   printf("Enter the total existed resource in each class:");

   for(k=1; k<=r; k++)
  {
     scanf("%d",&totext[k]);
     printf("Enter the allocated resources:");

   for(i=1; i<=n; i++)
  {
      for(k=1; k<=r; k++)
     {
         scanf("%d",&resalloc[i][k]);
         ("Enter the process making the new request:");
         scanf("%d",&p);
         printf("Enter the requested resource:");

   for(k=1; k<=r; k++)
          scanf("%d",&newreq[k]);
          printf("Enter the process which are n blocked or running:");

   for(i=1; i<=n; i++)
   {
      if(i!=p)
      {
         printf("process %d:\n",i+1);
         scanf("%d%d",&block[i],&run[i]);
      }
   }
   block[p]=0;
   run[p]=0;
   for(k=1; k<=r; k++)
   {
      j=0;

      for(i=1; i<=n; i++)
```

```
    {
       totalloc[k]=j+resalloc[i][k];
       j=totalloc[k];
    }
    }
for(i=1; i<=n; i++)
{
   if(block[i]==1||run[i]==1)
      active[i]=1;
   else
      active[i]=0;
}
for(k=1; k<=r; k++)
{
   resalloc[p][k]+=newreq[k];
   totalloc[k]+=newreq[k];
}
for(k=1; k<=r; k++)
{
   if(totext[k]-totalloc[k]<0)
   {
      u=1;
      break;
   }
}

if(u==0)
{
   for(k=1; k<=r; k++)
      simalloc[k]=totalloc[k];
   for(s=1; s<=n; s++)
      for(i=1; i<=n; i++)
      {
         if(active[i]==1)
         {
            j=0;
            for(k=1; k<=r; k++)
            {
               if((totext[k]-simalloc[k])<(max[i][k]-resalloc[i][k]))
               {
                  j=1;
                  break;
               }
            }
         }

         if(j==0)
         {
            active[i]=0;
    for(k=1; k<=r; k++)
```

```
                    simalloc[k]=resalloc[i][k];
                }
            }
        m=0;
        for(k=1; k<=r; k++)
            resreq[p][k]=newreq[k];
            printf("Deadlock willn't occur");
    }

    else
    {
        for(k=1; k<=r; k++)
        {
            resalloc[p][k]=newreq[k];
            totalloc[k]=newreq[k];
        }
        printf("Deadlock will occur");
    }
    return 0;
}
```

OUTPUT:

```
Enter the no of processes:4
Enter the no ofresource classes:3
Enter the total existed resource in each class:3 2 2
Enter the allocated resources:1 0 0 5 1 1 2 1 1 0 0 2
Enter the process making the new request:2
Enter the requested resource:1 1 2
Enter the process which are n blocked or running:process 2:
1 2
process 4:
1 0
process 5:
1 0
Deadlock will occur
```