****

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

****

```
#include<stdio.h>
struct proces
{
int allocation[3],max[3],need[3],finish;
}p[10];
int main()
{
int n,i,j,avail[3],work[3],flag,count=0,sequence[10],k=0;
printf("\nenter the no of processes:");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("\n enter the %d th process allocated resources:",i);
scanf("%d%d%d",&p[i].allocation[0],&p[i].allocation[1],&p[i].allocation[2]);
printf("\n enter the %d the process max resources:",i);
scanf("%d%d%d",&p[i].max[0],&p[i].max[1],&p[i].max[2]);
p[i].finish=0;
p[i].need[0]=p[i].max[0]-p[i].allocation[0];
p[i].need[1]=p[i].max[1]-p[i].allocation[1];
p[i].need[2]=p[i].max[2]-p[i].allocation[2];
}
printf("\n enter the available vector");
scanf("%d%d%d",&avail[0],&avail[1],&avail[2]);
for(i=0;i<3;i++)
work[i]=avail[i];
while(count!=n)
{
count=0;
for(i=0;i<n;i++)
{
flag=1;
if(p[i].finish==0)
```

```
if(p[i].need[0]<=work[0])
if(p[i].need[1]<=work[1])
if(p[i].need[2]<=work[2])
{
for(j=0;j<3;j++)
work[j]+=p[i].allocation[j];
p[i].finish=1;
sequence[k++]=i;
flag=0;
}
if(flag==1)
count++;
}
}
count=0;
for(i=0;i<n;i++)
if(p[i].finish==1)
count++;
printf("\n the safe sequence is:");
if(count==n)
for(i=0;i<k;i++)
printf("%d\t",sequence[i]);
else
printf("system is not in safe state \n\n");
return 0;
}
```