Program- Niyati Savant- Bankers Algorithm

CODE:

```
#include <stdio.h>
int main()
{
   int n, m, i, j, k, ind = 0;
   int alloc[10][10],max[10][10],need[10][10];;
   int avail[5],finish[5],ans[5];

   printf("Niyati's code for Bankers Algorithm \n");
   printf("Enter the Number of processes: ");
   scanf("%d",&n);
   printf("Enter the Number of resources: ");
   scanf("%d",&m);

   printf("Enter the Allocation Matrix:\n");
   for(i=0;i<n;i++)
   {
      printf("Enter for P%d: \n",i);
      for(j=0;j<m;j++)
         scanf("%d",&alloc[i][j]);
   }

   printf("Enter the Max Matrix:\n");
   for(i=0;i<n;i++)
   {
      printf("Enter for P%d: \n",i);
      for(j=0;j<m;j++)
         scanf("%d",&max[i][j]);
   }
```

```
printf("Enter available instances of \n:");
for(i=0;i<m;i++)
{
   printf("Resource %d: ",i+1);
   scanf("%d",&avail[i]);
}

for (k = 0; k < n; k++)
   finish[k] = 0;

for (i = 0; i < n; i++)
{
   for (j = 0; j < m; j++)
      need[i][j] = max[i][j] - alloc[i][j];
}
int y = 0;
for (k = 0; k < n; k++)
{
   for (i = 0; i < n; i++)
   {
      if (finish[i] == 0)
      {
         int flag = 0;
         for (j = 0; j < m; j++)
         {
            if (need[i][j] > avail[j])
            {
               flag = 1;
               break;
            }
         }
```

```
            if (flag == 0)
            {
               ans[ind++] = i;
               for (y = 0; y < m; y++)
                  avail[y] += alloc[i][y];
               finish[i] = 1;
            }
         }
      }

   int flag = 1;
   for (int i = 0; i < n; i++)
   {
      if (finish[i] == 0)
      {
         flag = 0;
         printf("The following system is not safe");
         break;
      }
   }
   if (flag == 1)
   {
      printf("Following is the SAFE Sequence\n");
      for (i = 0; i < n - 1; i++)
         printf(" P%d ->", ans[i]);
      printf(" P%d", ans[n - 1]);
   }
   return (0);
}
```

OUTPUT:

Niyati’s code for Bankers Algorithm

Enter the Number of processes: 5

Enter the Number of resources: 3

Enter the Allocation Matrix:

Enter for P0:

0 1 0

Enter for P1:

2 0 0

Enter for P2:

3 0 2

Enter for P3:

2 1 1

Enter for P4:

0 0 2

Enter the Max Matrix:

Enter for P0:

7 5 3

Enter for P1:

3 2 2

Enter for P2:

9 0 2

Enter for P3:

2 2 2

Enter for P4:

4 3 3

Enter available instances of

:Resource 1: 3

Resource 2: 3

Resource 3: 2

Following is the SAFE Sequence

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