OS Assignment 7

Name: Yash Oswal
Div: B   Roll no: 38
SRN: 201901226

1. BANKER’S

ALGORITHM.

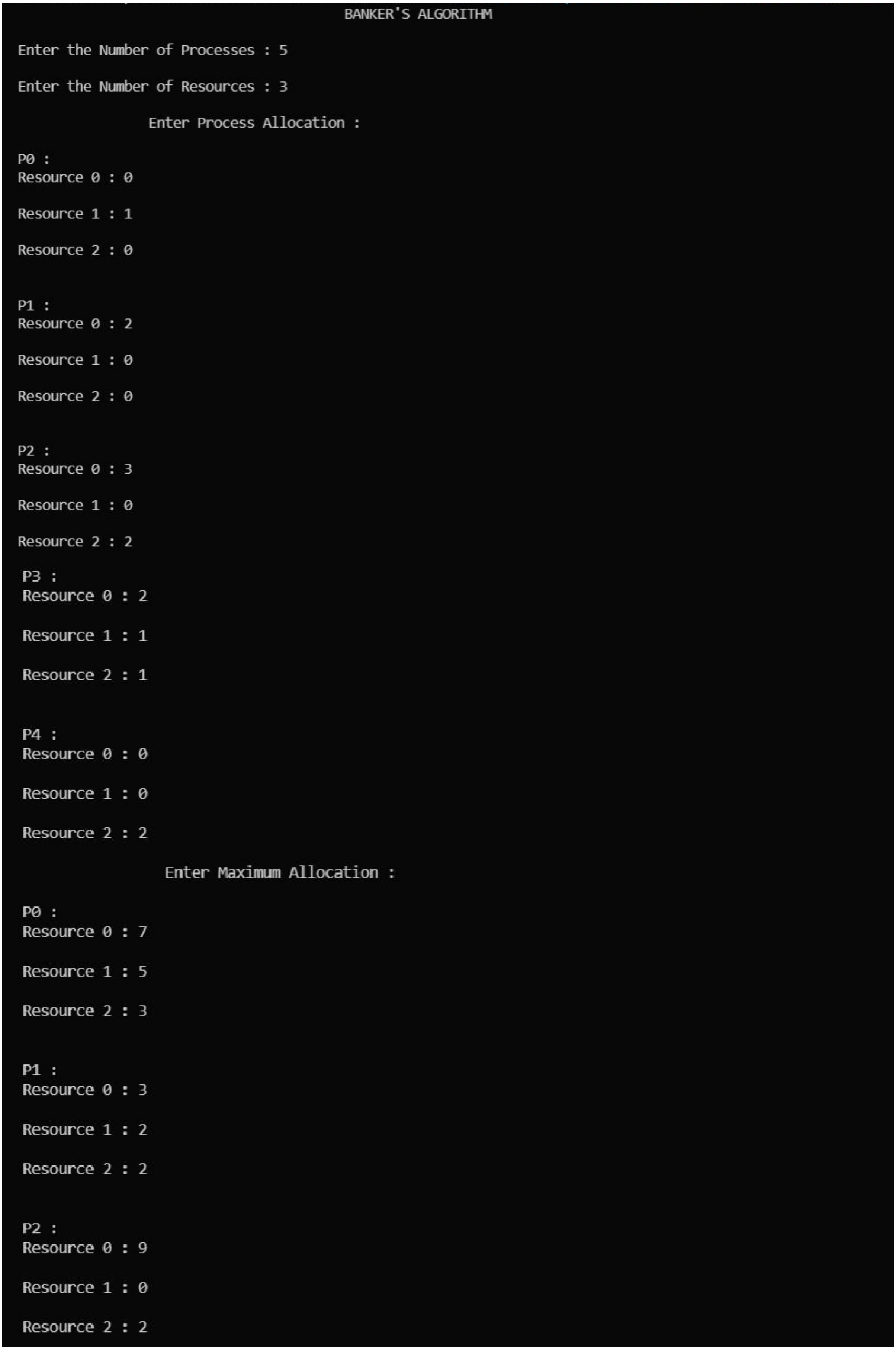
Code -

```
#include
<iostream
> using
namespac
e std; int
main()
{
  int i, j, k, n, m, y = 0;
  cout << "\t\t\t\t\tBANKER'S ALGORITHM";
  cout << "\n\nEnter the Number of
  Processes : "; cin >> n;
  cout << "\nEnter the Number of
  Resources : "; cin >> m;
  int alloc[n][m], max[n][m], avail[m];
  int f[n], ans[n], ind = 0, need[n][m];
  cout << "\n\t\tEnter Process
  Allocation : "; for (i = 0; i < n;
  i++)
  {
    cout << "\n\nP"
    << i << " : "; for
    (j = 0; j < m;
    j++)
    {
      cout << "\nResource "
      << j << " : "; cin >>
      alloc[i][j];
    }
  }
  cout << "\n\t\tEnter Maximum
  Allocation : "; for (i = 0; i < n;
  i++)
  {
    cout << "\n\nP"
    << i << " : "; for
    (j = 0; j < m;
    j++)
    {
        cout << "\nResource "
        << j << " : ";cin >>
        max[i][j];
    }
}
cout << "\n\t\tEnter Available
Resources : ";for (i = 0; i < m;
i++)
```

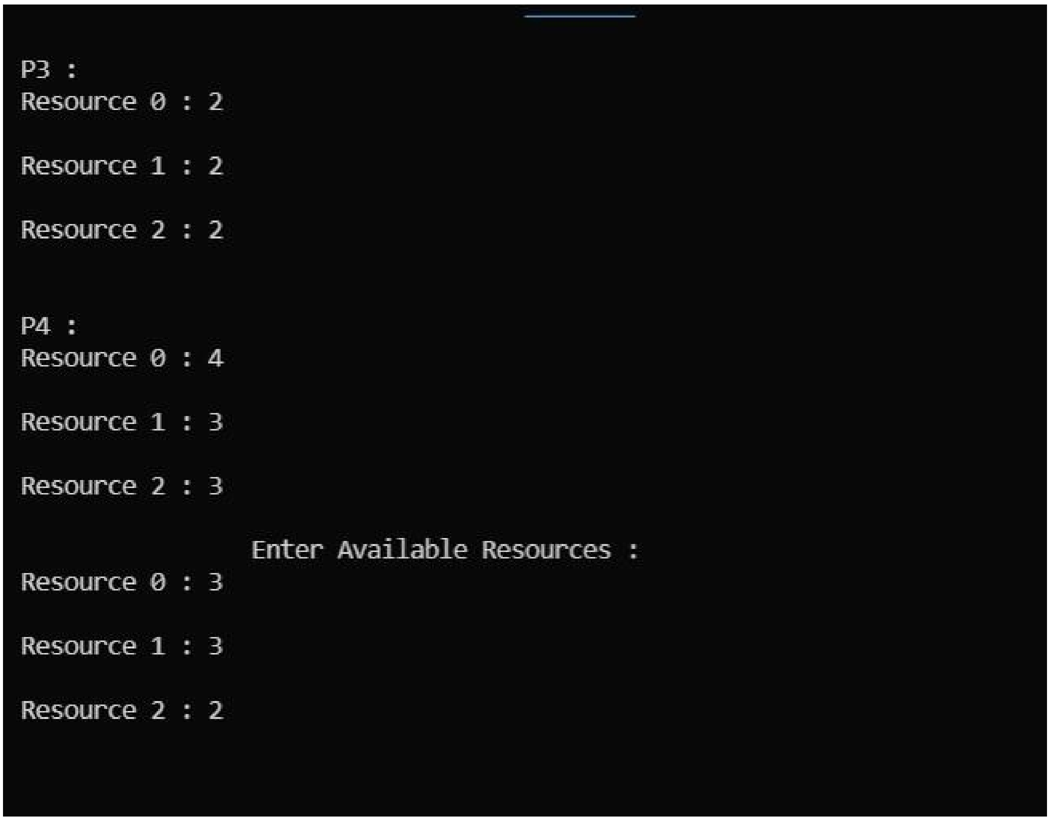
```
{
   cout << "\nResource " << i << " :
   ";cin >> avail[i];
}
cout << "\n\n\n\t\t\tProcess Allocation :
\n\n";for (i = 0; i < m; i++)
{
   cout << "\t\tR" << i;
}
for (i = 0; i < n; i++)
{
   cout << "\nP" << i;
   for (j = 0; j < m;
   j++)
   {
      cout << "\t\t" << alloc[i][j];
   }
}
cout << "\n\n\t\t\tMaximum Allocation :
\n\n";for (i = 0; i < m; i++)
{
   cout << "\t\tR" << i;
}
for (i = 0; i < n; i++)
{
   cout << "\nP" << i;
   for (j = 0; j < m;
   j++)
   {
      cout << "\t\t" << max[i][j];
   }
}
cout << "\n\n\t\t\tAvailable Resources :
\n\n";for (i = 0; i < m; i++)
{
   cout << "\t\tR" << i;
}
cout << "\n";
for (i = 0; i < m; i++)
{
   cout << "\t\t" << avail[i];
}
y = 0;
for (k = 0; k < n; k++)
{
   f[k] = 0;
}
for (i = 0; i < n; i++)
{
   for (j = 0; j < m; j++)
      need[i][j] = max[i][j] - alloc[i][j];
}
for (k = 0; k < n; k++)
```

```
  {
    for (i = 0; i < n; i++)
    {
      if (f[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];
          f[i] = 1;
        }
      }
    }
  }
  cout << "\n\nSAFE PROCESS SEQUENCE : \n";
  for (i = 0; i < n - 1; i++)
    cout << " P" << ans[i] << " -
  >"; cout << " P" << ans[n - 1]
  << endl;return (0);
}
```

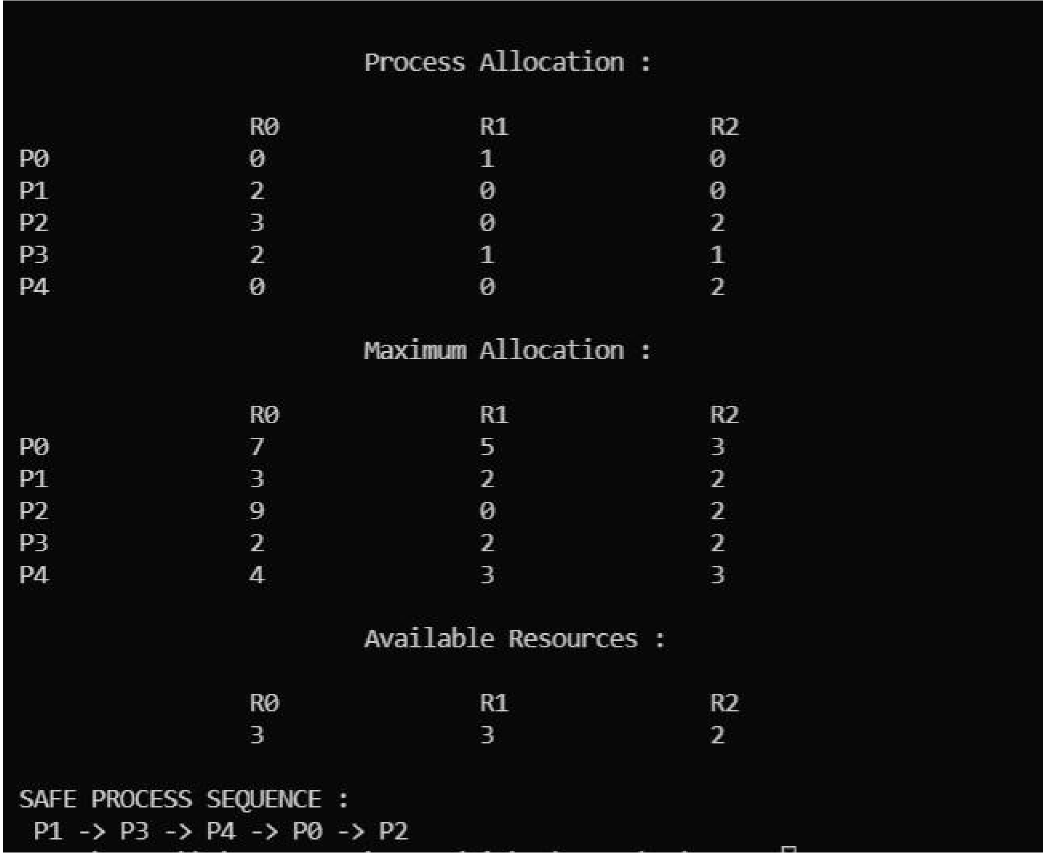
Output -



```
                                          BANKER'S ALGORITHM

Enter the Number of Processes : 5

Enter the Number of Resources : 3

                Enter Process Allocation :

P0 :
Resource 0 : 0

Resource 1 : 1

Resource 2 : 0


P1 :
Resource 0 : 2

Resource 1 : 0

Resource 2 : 0


P2 :
Resource 0 : 3

Resource 1 : 0

Resource 2 : 2

P3 :
Resource 0 : 2

Resource 1 : 1

Resource 2 : 1


P4 :
Resource 0 : 0

Resource 1 : 0

Resource 2 : 2

               Enter Maximum Allocation :

P0 :
Resource 0 : 7

Resource 1 : 5

Resource 2 : 3


P1 :
Resource 0 : 3

Resource 1 : 2

Resource 2 : 2


P2 :
Resource 0 : 9

Resource 1 : 0

Resource 2 : 2
```



```
P3 :
Resource 0 : 2

Resource 1 : 2

Resource 2 : 2


P4 :
Resource 0 : 4

Resource 1 : 3

Resource 2 : 3

                Enter Available Resources :
Resource 0 : 3

Resource 1 : 3

Resource 2 : 2
```



```
                        Process Allocation :

                R0              R1              R2
P0              0               1               0
P1              2               0               0
P2              3               0               2
P3              2               1               1
P4              0               0               2

                        Maximum Allocation :

                R0              R1              R2
P0              7               5               3
P1              3               2               2
P2              9               0               2
P3              2               2               2
P4              4               3               3

                        Available Resources :

                R0              R1              R2
                3               3               2

SAFE PROCESS SEQUENCE :
 P1 -> P3 -> P4 -> P0 -> P2
```