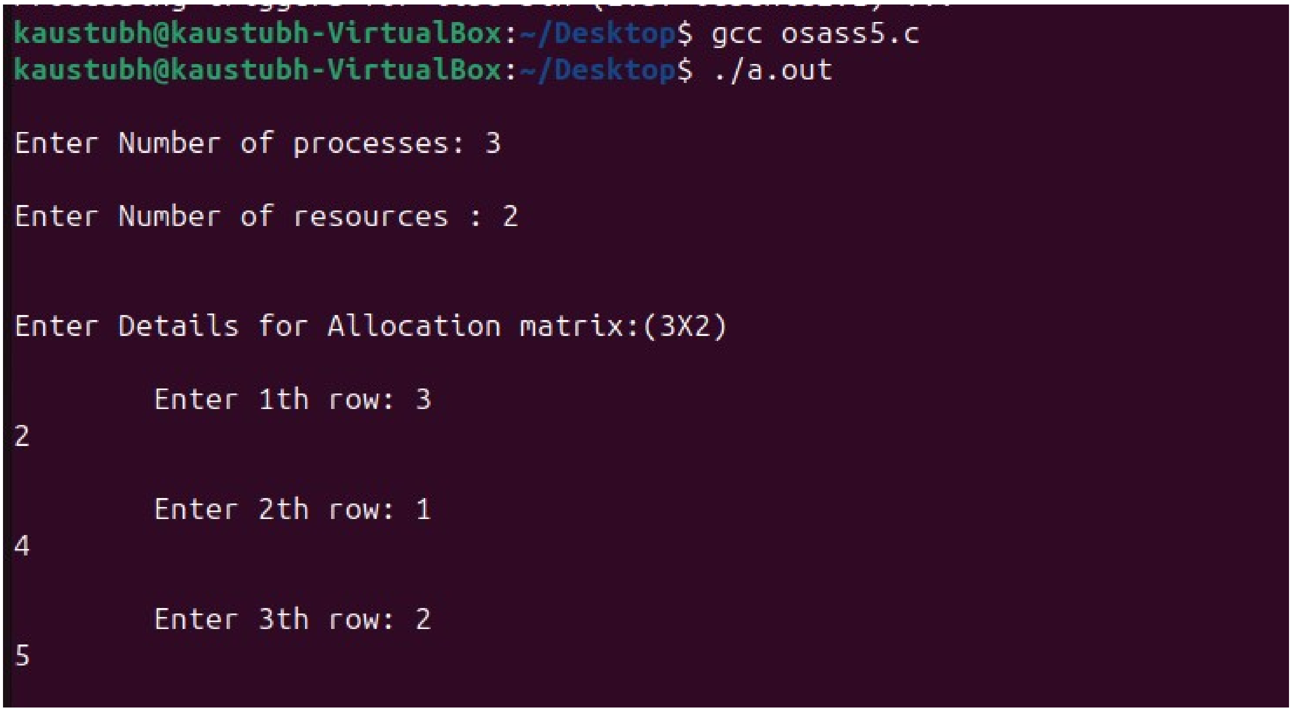
NAME: Kaustubh Gajanan Indulkar
ROLL NO: 35026 (25027)
CLASS: TE(IT)-A

ASSIGNMENT 5

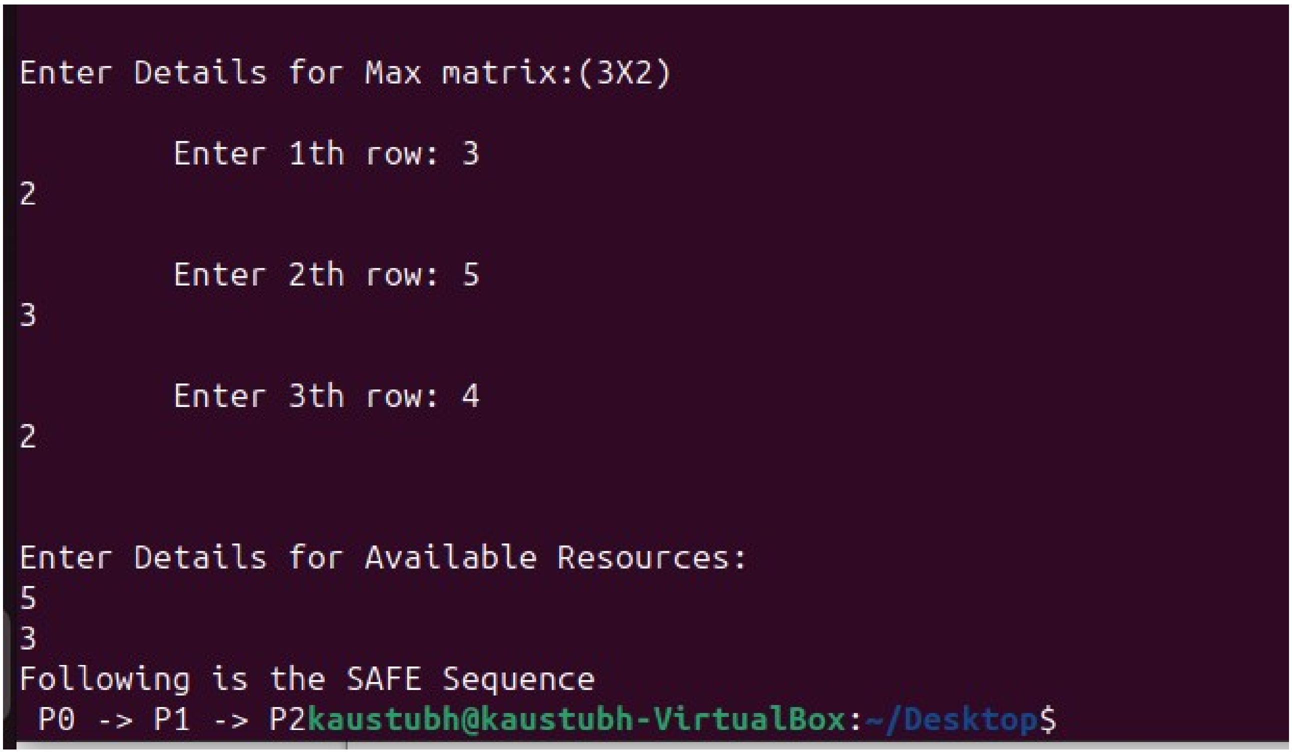
Problem Statement: Implement the C program for Deadlock Avoidance Algorithm: Bankers Algorithm.

```
#include <stdio.h>
int main()
{
    // P0, P1, P2, P3, P4 are the Process names here
    int n, m, i, j, k;
    printf("\nEnter Number of processes: ");
    scanf("%d", &n);

    printf("\nEnter Number of resources : ");
    scanf("%d", &m);

    int alloc[n][m];
    int max[n][m];

    printf("\n\nEnter Details for Allocation matrix:(%dX%d) \n", n, m);
    for (int i = 0; i < n; i++)
    {
        printf("\n\tEnter %dth row: ", i + 1);
        for (int j = 0; j < m; j++)
        {
            scanf("%d", &alloc[i][j]);
        }
    }
    printf("\n\nEnter Details for Max matrix:(%dX%d) \n", n, m);
    for (int i = 0; i < n; i++)
    {
        printf("\n\tEnter %dth row: ", i + 1);
        for (int j = 0; j < m; j++)
        {
            scanf("%d", &max[i][j]);
        }
    }

    int avail[m]; // Available Resources
    printf("\n\nEnter Details for Available Resources:\n");
    for (int j = 0; j < m; j++)
    {
        scanf("%d", &avail[j]);
    }

    int f[n], ans[n], ind = 0;
    for (k = 0; k < n; k++)
    {
        f[k] = 0;
    }
    int need[n][m];
    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 < m; 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;
                }
            }
        }
    }

    int flag = 1;

    for (int i = 0; i < n; i++)
    {
        if (f[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);
}
```



```
kaustubh@kaustubh-VirtualBox:~/Desktop$ gcc osass5.c
kaustubh@kaustubh-VirtualBox:~/Desktop$ ./a.out

Enter Number of processes: 3

Enter Number of resources : 2


Enter Details for Allocation matrix:(3X2)

        Enter 1th row: 3
2

        Enter 2th row: 1
4

        Enter 3th row: 2
5
```

NAME: Kaustubh Gajanan Indulkar
ROLL NO: 35026 (25027)
CLASS: TE(IT)-A



```
Enter Details for Max matrix:(3X2)

        Enter 1th row: 3
2

        Enter 2th row: 5
3

        Enter 3th row: 4
2


Enter Details for Available Resources:
5
3
Following is the SAFE Sequence
 P0 -> P1 -> P2kaustubh@kaustubh-VirtualBox:~/Desktop$
```