# Name – Manish Kumar

# Enrollment No - 2020ITB007

# 

**Deadlocks**

Consider  
a system that contains five processes P1, P2, P3, P4, P5 and the three  
resource types A, B and C. Following are the resource types: A has 10, B has 5,  
and resource type C has 7 instances.   
  
Implement  
the Bankers algorithm using C Program.   
  
1.       
Show the  
Available, Max, Allocation and Need matrices.   
  
2.       
Determine  
if the system is safe or not.  
  
3.       
Check  
your code if the resource request (1, 0, 0) for process P1 can the system  
accept this request immediately.

#include <stdio.h>

int main()

{

    int alloc[5][3], max[5][3], avail[3] = {10, 5, 7}, need[5][3];

    for (int i = 0; i < 5; i++)

    {

        printf("Enter max values for P%d : ", i + 1);

        for (int j = 0; j < 3; j++)

        {

            scanf("%d", &max[i][j]);

        }

        printf("\n");

    }

    printf("\n");

    for (int i = 0; i < 5; i++)

    {

        printf("Enter allocation values for P%d : ", i + 1);

        for (int j = 0; j < 3; j++)

        {

            scanf("%d", &alloc[i][j]);

        }

        printf("\n");

    }

    printf("\nAvailable array\n\n");

    printf("A  B C\n");

    for (int j = 0; j < 3; j++)

    {

        printf("%d ", avail[j]);

    }

    printf("\n\nMax Matrix\n\n");

    for (int i = 0; i < 5; i++)

    {

        printf("P%d :", i + 1);

        for (int j = 0; j < 3; j++)

        {

            printf("%d ", max[i][j]);

        }

        printf("\n");

    }

    printf("\nAllocation Matrix\n\n");

    for (int i = 0; i < 5; i++)

    {

        printf("P%d :", i + 1);

        for (int j = 0; j < 3; j++)

        {

            printf("%d ", alloc[i][j]);

        }

        printf("\n");

    }

    printf("\nNeed Matrix\n\n");

    for (int i = 0; i < 5; i++)

    {

        printf("P%d :", i + 1);

        for (int j = 0; j < 3; j++)

        {

            need[i][j] = max[i][j] - alloc[i][j];

            printf("%d ", need[i][j]);

        }

        printf("\n");

    }

    int fin[5], work[3];

    for (int i = 0; i < 5; i++)

        fin[i] = 0;

    for (int i = 0; i < 3; i++)

        work[i] = avail[i];

    int ind = 0, ans[5];

    for (int k = 0; k < 5; k++)

    {

        for (int i = 0; i < 5; i++)

        {

            if (fin[i] == 0)

            {

                int flag = 1;

                for (int j = 0; j < 3; j++)

                {

                    if (need[i][j] > work[j])

                    {

                        flag = 0;

                        break;

                    }

                }

                if (flag == 1)

                {

                    for (int l = 0; l < 3; l++)

                        work[l] = work[l] + alloc[i][l];

                    fin[i] = 1;

                    ans[ind] = i;

                    ind++;

                }

            }

        }

    }

    int check = 1;

    for (int i = 0; i < 5; i++)

    {

        if (fin[i] == 0)

        {

            printf("\n\nSystem is not safe");

            check = 0;

            break;

        }

    }

    if (check)

    {

        printf("\n\nSystem is safe\nOrder of Processes : ");

        for (int i = 0; i < 5; i++)

        {

            printf("P%d", ans[i] + 1);

            if (i < 4)

                printf(" -> ");

        }

    }

    int req[3] = {1, 0, 0};

    printf("\n\nResource request for Process P1 : 1 0 0");

    for (int j = 0; j < 3; j++)

    {

        if (need[0][j] < req[j])

        {

            printf("\n\nProcess has exceeded its max claim");

            check = 0;

            break;

        }

    }

    if (check)

    {

        for (int i = 0; i < 3; i++)

        {

            need[0][i] -= req[i];

            avail[i] -= req[i];

            alloc[0][i] += req[i];

        }

        for (int i = 0; i < 5; i++)

        {

            fin[i] = 0;

        }

        ind = 0;

        for (int k = 0; k < 5; k++)

        {

            for (int i = 0; i < 5; i++)

            {

                if (fin[i] == 0)

                {

                    int flag = 1;

                    for (int j = 0; j < 3; j++)

                    {

                        if (need[i][j] > avail[j])

                        {

                            flag = 0;

                            break;

                        }

                    }

                    if (flag == 1)

                    {

                        for (int l = 0; l < 3; l++)

                            avail[l] = avail[l] + alloc[i][l];

                        fin[i] = 1;

                        ans[ind] = i;

                        ind++;

                    }

                }

            }

        }

        int flag = 1;

        for (int i = 0; i < 5; i++)

        {

            if (fin[i] == 0)

            {

                printf("\n\nSystem is not safe after request");

                flag = 0;

                break;

            }

        }

        if (flag)

        {

            printf("\n\nSystem is safe after request\nOrder of Processes : ");

            for (int i = 0; i < 5; i++)

            {

                printf("P%d", ans[i] + 1);

                if (i < 4)

                    printf(" -> ");

            }

        }

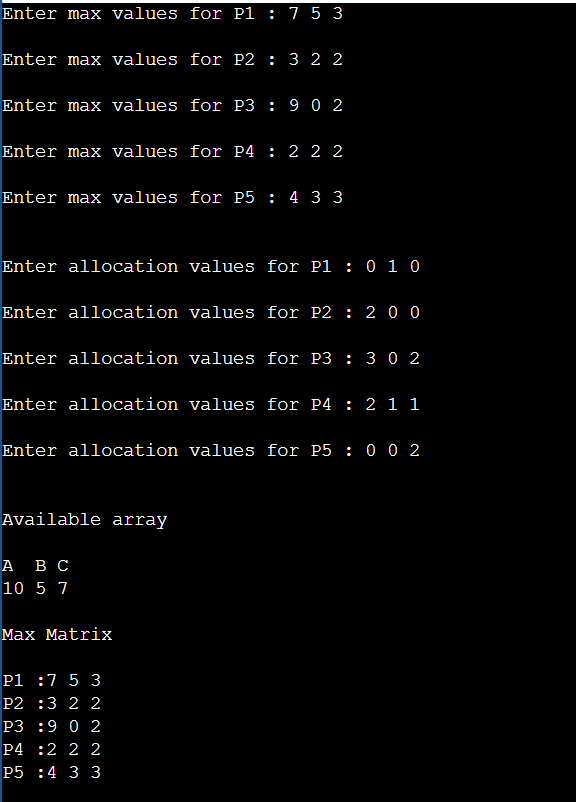
    }

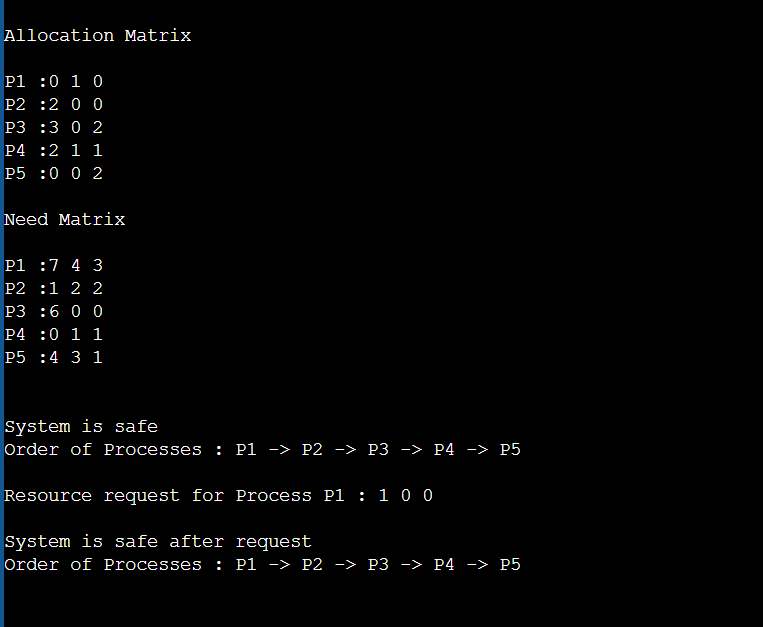
    printf("\n\n");

    return 0;

}

**OUTPUT**

****

****