**OS\_LAB\_12\_Assignment**

**CE\_054**

Aim :- Implementation of banker’s Algorithm in C language.

Code :-

// Author : Dhruv B Kakadiya

#include <stdio.h>

int main()

{

    int processes, res, i, j, k;

    printf("Enter the number of Resources and processes : -\n");

    scanf("%d%d", &res, &processes);

    int allocation[processes][res];

    printf("Enter the allocation matrix : -\n");

    for (i = 0 ; i < processes ; i++)

    {

        for (j = 0 ; j < res ; j++)

        {

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

        }

    }

    int maximum\_need[processes][res];

    printf("Enter the max matrix : -\n");

    for (i = 0 ; i < processes ; i++)

    {

        for (j = 0 ; j < res ; j++)

        {

            scanf("%d", &maximum\_need[i][j]);

        }

    }

    int available[res];

    printf("Enter the number of needed resources : -\n");

    for (i = 0 ; i < res ; i++)

    {

        scanf("%d", &available[i]);

    }

    int f[processes], ans[processes], ind = 0;

    for (k = 0 ; k < processes ; k++)

    {

        f[k] = 0;

    }

    int need[processes][res];

    for (i = 0 ; i < processes ; i++)

    {

        for (j = 0 ; j < res ; j++)

            need[i][j] = maximum\_need[i][j] - allocation[i][j];

    }

    int y = 0;

    for (k = 0 ; k < processes ; k++)

    {

        for (i = 0 ; i < processes ; i++)

        {

            if (f[i] == 0)

            {

                int flag = 0;

                for (j = 0; j < res; j++)

                {

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

                    {

                        flag = 1;

                        break;

                    }

                }

                if (flag == 0)

                {

                    ans[ind++] = i;

                    for (y = 0 ; y < res ; y++)

                        available[y] += allocation[i][y];

                    f[i] = 1;

                }

            }

        }

    }

    printf("\nFollowing is the SAFE Sequence\n");

    for (i = 0; i < processes - 1; i++)

        printf(" P%d ->", ans[i]);

    printf(" P%d\n\n", ans[processes - 1]);

    return (0);

}

Output :-

