Banker Algorithm 33185 OS

```
#include<stdio.h>

#include<conio.h>

#define true 1

#define false 0

int available[10], allocation[10][10], max[10][10], need[10][10], work[10],
finish[10], maxres[10], safe[10], req[10], m, n;

int find()

{

        int i, j;

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

        {

                if (finish[i] == false)

                {

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

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

                        if (j == m)

                        {

                                finish[i] = true;

                                return i;

                        }

                }

        }

        return -1;

}


int issafe()

{

        int i = 0, j, k = 0, cnt = n;

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

                work[j] = available[j];

        for (j = 0; j < m; j++)
```

```
            finish[i] = false;
        while (cnt > 0)
        {
            for (i = 0; i < n; i++)
            {
                i = find();
                if (i == -1)
                {
                    printf("\nThe system is in unsafe state");
                    return 0;
                }
                for (j = 0; j < m; j++)
                    work[j] += allocation[i][j];
                safe[k++] = i;
                cnt--;
            }
        }
        if (finish[i - 1] == false)
        {
            printf("\nThe system is in unsafe state");
            return 0;
        }
        printf("\nThe system is in safe state, safe sequence: ");
        for (i = 0; i < n; i++)
            printf("P%d, ", safe[i]);
        return 0;
}
int main()
{
        int i, j, sum;
        char ch;
        printf("\nEnter the number of processes and the number of resources:\n");
```

```c
scanf("%d%d", &n, &m);
printf("\nEnter maximum instances of resources\n");
for (j = 0; j < m; j++)
{
	scanf("%d", &maxres[j]);
	available[j] = maxres[j];
}
printf("\nEnter the Allocated Matrix:\n");
for (i = 0; i < n; i++)
{
	for (j = 0; j < m; j++)
		scanf("%d", &allocation[i][j]);
}
printf("\nEnter the Max Matrix:\n");
for (i = 0; i < n; i++)
{
	for (j = 0; j < m; j++)
	{
		scanf("%d", &max[i][j]);
		need[i][j] = max[i][j] - allocation[i][j];
	}
}
printf("\nThe Matrix is:\n");
for (i = 0; i < n; i++)
{
	for (j = 0; j < m; j++)
		printf("%d ", need[i][j]);
	printf("\n");
}
for (j = 0; j < m; j++)
{
	sum = 0;
```

```
                for (i = 0; i < n; i++)
                        sum += allocation[i][j];
                available[j] -= sum;
        }
        issafe();
}
```

Output:

```
Enter the number of processes and the number of resources:
5 3

Enter maximum instances of resources
3
2
4

Enter the Allocated Matrix:
2
5
4
3
5
6
4
7
5
3
5
7
5
4
```

```
7

Enter the Max Matrix:
4
6
5
4
6
3
6
3
6
7
4
7
6
8
6

The Matrix is:
2 1 1
1 1 -3
2 -4 1
4 -1 0
1 4 -1

The system is in unsafe state
```