

LAB 08

Implement the above code and paste the screen shot of the output.

CODE:

```
#include <stdio.h>

#include <conio.h>

int max[100][100];
int alloc[100][100];
int need[100][100];
int avail[100];

int n, r;

void input();
void show();
void cal();

int main() {
    int i, j;

    printf("***** Deadlock Detection Algo *****\n");

    input();

    show();

    cal();

    getch();

    return 0;
}

void input() {
    int i, j;

    printf("Enter the no of Processes: ");

    scanf("%d", &n);

    printf("Enter the no of resource instances: ");
```

```
scanf("%d", &r);

printf("Enter the Max Matrix\n");

for (i = 0; i < n; i++) {
    for (j = 0; j < r; j++) {
        scanf("%d", &max[i][j]);
    }
}

printf("Enter the Allocation Matrix\n");

for (i = 0; i < n; i++) {
    for (j = 0; j < r; j++) {
        scanf("%d", &alloc[i][j]);
    }
}

printf("Enter the Available Resources\n");

for (j = 0; j < r; j++) {
    scanf("%d", &avail[j]);
}

}

void show() {
    int i, j;

    printf("Process\tAllocation\tMax\t\tAvailable\n");

    for (i = 0; i < n; i++) {
        printf("P%d\t", i + 1);
        for (j = 0; j < r; j++) {
            printf("%d ", alloc[i][j]);
        }
        printf("\t\t");
        for (j = 0; j < r; j++) {
```

```
        printf("%d ", max[i][j]);
    }
    if (i == 0) {
        printf("\t\t");
        for (j = 0; j < r; j++) {
            printf("%d ", avail[j]);
        }
    }
    printf("\n");
}

void cal() {
    int finish[100], flag = 1, dead[100], safe[100];
    int i, j, k, c1 = 0;
    // Initialize finish array
    for (i = 0; i < n; i++) {
        finish[i] = 0;
    }

    // Calculate Need Matrix
    for (i = 0; i < n; i++) {
        for (j = 0; j < r; j++) {
            need[i][j] = max[i][j] - alloc[i][j];
        }
    }
    while (flag) {
        flag = 0;
        for (i = 0; i < n; i++) {
            int count = 0;
```

```
        if (!finish[i]) {
            for (j = 0; j < r; j++) {
                if (need[i][j] <= avail[j]) {
                    count++;
                }
            }
            if (count == r) {
                for (k = 0; k < r; k++) {
                    avail[k] += alloc[i][k];
                }
                finish[i] = 1;
                flag = 1;
            }
        }
    }

    // Check for deadlock
    int deadlockExists = 0;
    int deadCount = 0;

    for (i = 0; i < n; i++) {
        if (!finish[i]) {
            dead[deadCount++] = i;
            deadlockExists = 1;
        }
    }

    if (deadlockExists) {
        printf("\n\nSystem is in Deadlock and the Deadlocked processes are:\n");
        for (i = 0; i < deadCount; i++) {
```

```

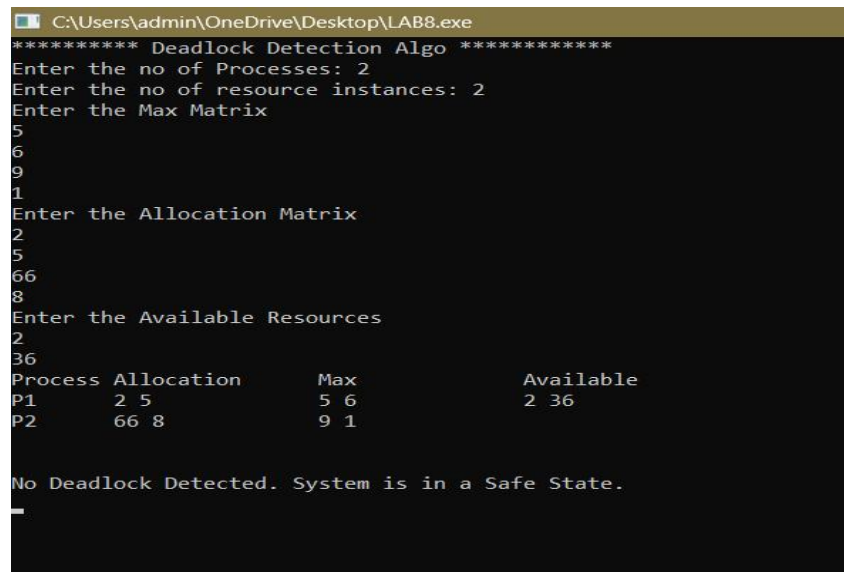
        printf("P%d\t", dead[i]);
    }

    printf("\n");
} else {

    printf("\n\nNo Deadlock Detected. System is in a Safe State.\n");
}
}

```

OUTPUT:



```

C:\Users\admin\OneDrive\Desktop\LAB8.exe
***** Deadlock Detection Algo *****
Enter the no of Processes: 2
Enter the no of resource instances: 2
Enter the Max Matrix
5
6
9
1
Enter the Allocation Matrix
2
5
66
8
Enter the Available Resources
2
36
Process Allocation      Max      Available
P1      2 5      5 6      2 36
P2      66 8      9 1
No Deadlock Detected. System is in a Safe State.

```