

NAME: LAIBA NADEEM

ROLL NUMBER: DT-22028

LAB No: 8

DEADLOCK

Exercise:

- 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 number of Processes: ");  
    scanf("%d", &n);  
    printf("Enter the number 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\t Allocation\t Max\t Available\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");
        for(j = 0; j < r; j++)
        {
            printf("%d ", max[i][j]);
        }
        printf("\t");
        if(i == 0)
        {
            for(j = 0; j < r; j++)
                printf("%d ", avail[j]);
        }
        printf("\n");
    }
}

void cal()
{
    int finish[100], temp, need[100][100], flag = 1, k, c1 = 0;
    int dead[100];
    int safe[100];
    int i, j;

```

```

for(i = 0; i < n; i++)
{
    finish[i] = 0;
}

// Find 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 c = 0;
        for(j = 0; j < r; j++)
        {
            if((finish[i] == 0) && (need[i][j] <= avail[j]))
            {
                c++;
                if(c == r)
                {
                    for(k = 0; k < r; k++)
                    {
                        avail[k] += alloc[i][k];
                    }
                }
            }
        }
    }
}

```

```

        finish[i] = 1;
        flag = 1;
    }
}

}

}

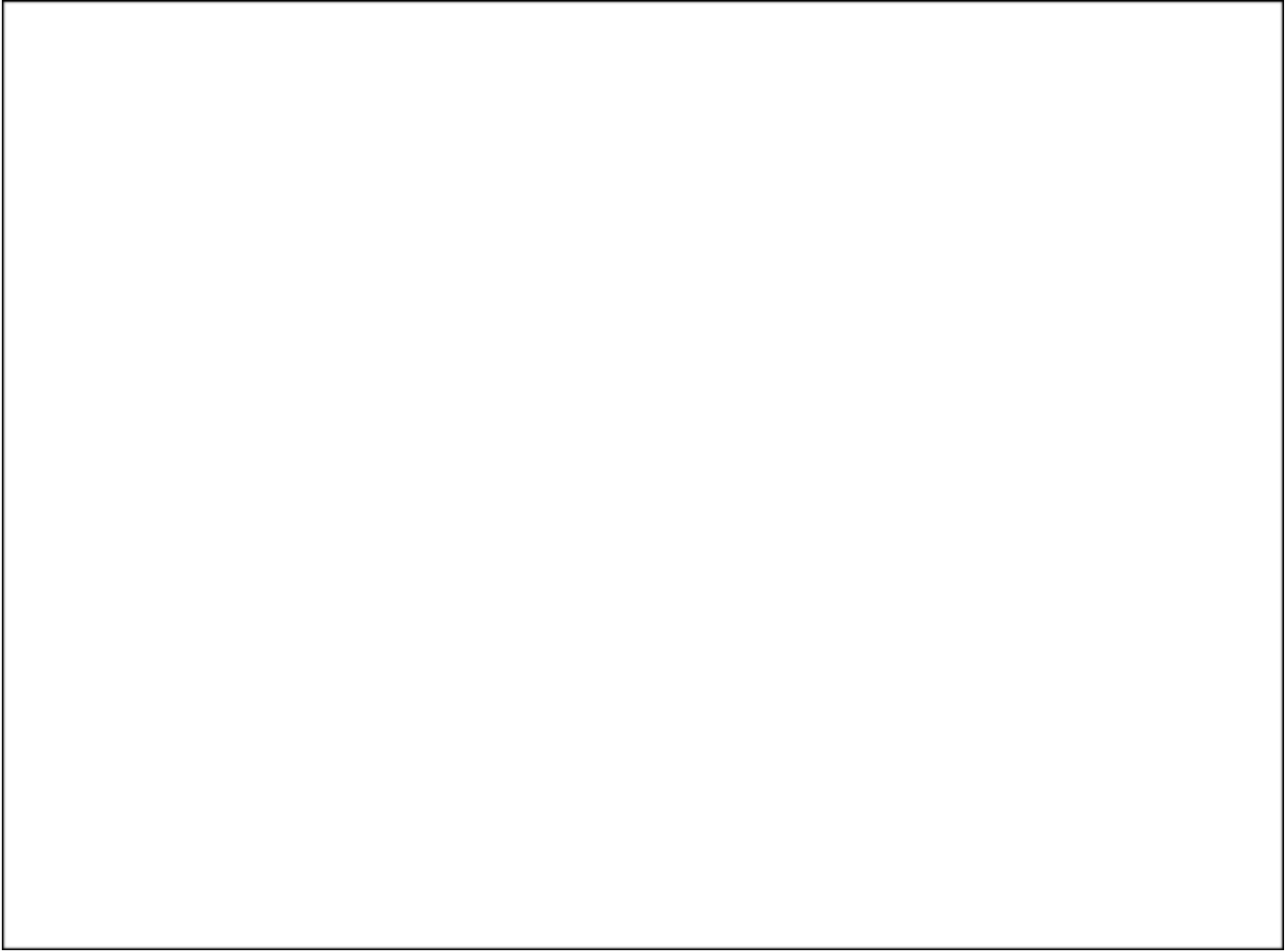
j = 0;
flag = 0;
for(i = 0; i < n; i++)
{
    if(finish[i] == 0)
    {
        dead[j] = i;
        j++;
        flag = 1;
    }
}

if(flag == 1)
{
    printf("\n\nSystem is in Deadlock, and the Deadlock
processes are:\n");
    for(i = 0; i < j; i++)
    {
        printf("P%d\t", dead[i]);
    }
    printf("\n");
}
else
{

```

```
        printf("\nSystem is not in Deadlock\n");  
    }  
}
```

OUTPUT



***** Deadlock Detection Algo *****

Enter the number of Processes: 3

Enter the number of resource instances: 3

Enter the Max Matrix:

7 5 3

3 2 2

9 0 2

Enter the Allocation Matrix:

0 1 0

2 0 0

3 0 2

Enter the Available Resources:

3 3 2

Process	Allocation	Max	Available
P1	0 1 0	7 5 3	3 3 2
P2	2 0 0	3 2 2	
P3	3 0 2	9 0 2	

System is in Deadlock, and the Deadlock processes are:

P0 P2