WEEK 8

Write a C program to simulate deadlock detection

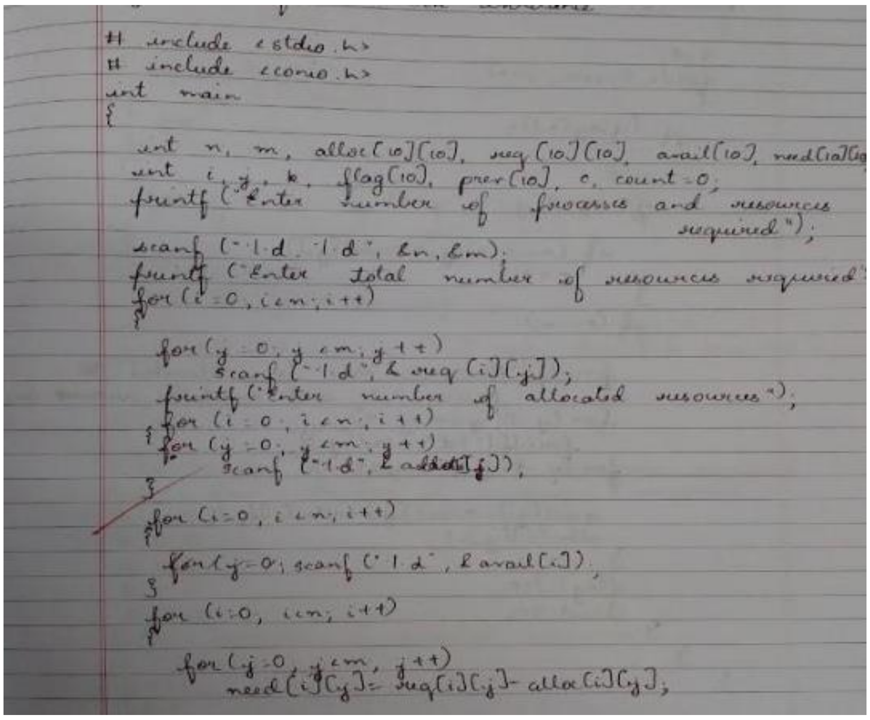
CODE :

```
#include <stdio.h>
int main()
 {
   int n, m, all[10][10], req[10][10], ava[10], need[10][10];
   int i, j, k, flag[10], prev[10], c, count = 0;
   printf("Enter number of processes and number of resources required \n");
   scanf("%d %d", &n, &m);
   printf("Enter total number of required resources %d for each process\n", n);
   for (i = 0; i < n; i++)
         for (j = 0; j < m; j++)
         scanf("%d", &req[i][j]);
   printf("Enter number of allocated resources %d for each process\n", n);
   for (i = 0; i < n; i++)
         for (j = 0; j < m; j++)
         scanf("%d", &all[i][j]);
   printf("Enter number of available resources \n");
   for (i = 0; i < m; i++)
         scanf("%d", &ava[i]);
   for (i = 0; i < n; i++)
         for (j = 0; j < m; j++)
         need[i][j] = req[i][j] - all[i][j];
   for (i = 0; i < n; i++)
```

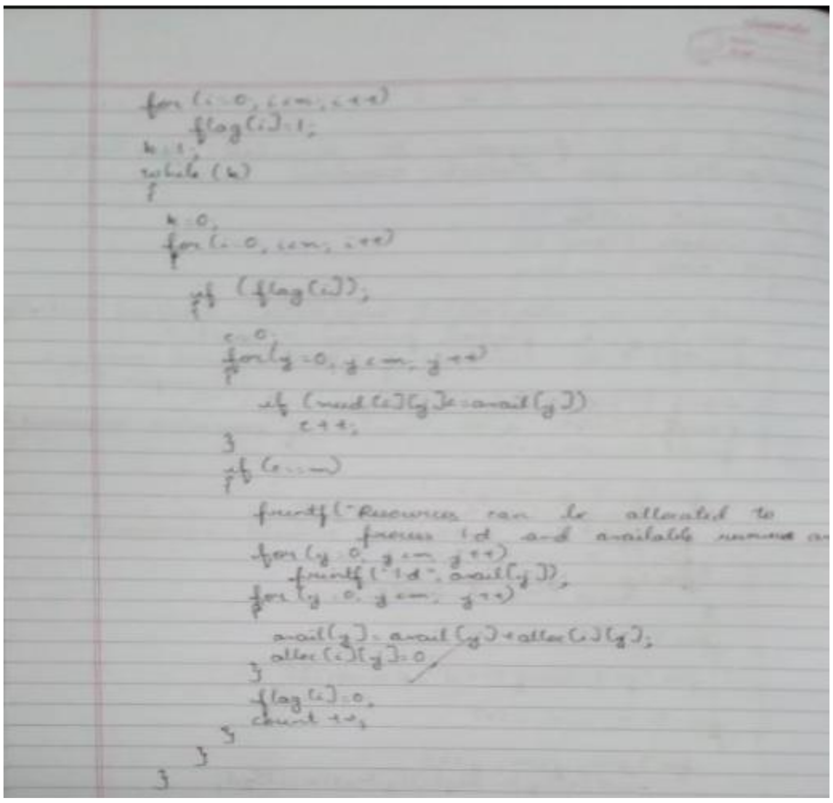
```
        flag[i] = 1;
    k = 1;
    while (k) {
      k = 0;
        for (i = 0; i < n; i++) {
        if (flag[i]) {
              c = 0;
              for (j = 0; j < m; j++) {
              if (need[i][j] <= ava[j])
                c++;
        }
              if (c == m) {
              for (j = 0; j < m; j++) {
              }
              for (j = 0; j < m; j++) {
              ava[j] += all[i][j];
              all[i][j] = 0;
              }
              flag[i] = 0;
              count++;
              }
        }
        }
        for (i = 0; i < n; i++) {
        if (flag[i] != prev[i]) {
              k = 1;
              break;
```

```
          }
          }
          for (i = 0; i < n; i++) {
          prev[i] = flag[i];
          }
    }
    if (count == n) {
          printf("\nNo deadlock");
    } else {
          printf("\nDeadlock occurred \n");
    }
    return 0;
}
```

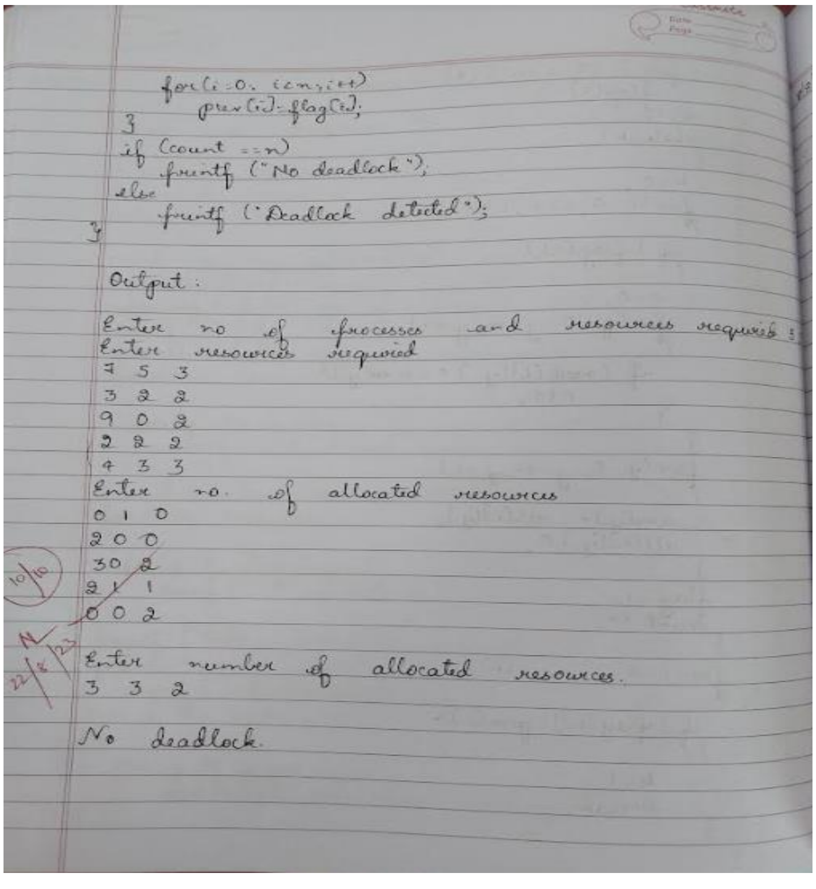
OBSERVATION :



```
# include <stdio.h>
# include <conio.h>
int main
{
   int n, m, alloc[10][10], req[10][10], avail[10], need[10][10
   int i, j, k, flag[10], prev[10], c, count=0;
   printf("Enter number of processes and resources
                                         required");
   scanf("%d %d", &n, &m);
   printf("Enter total number of resources required
   for(i=0, i<n; i++)
   {
     for(j=0, j<m; j++)
          scanf("%d", &req[i][j]);
     printf("Enter number of allocated resources");
     for (i=0, i<n; i++)
    {for (j=0, j<m; j++)
     {    scanf("%d", &alloc[i][j]),
    }
    for (i=0, i<n; i++)
    {
      for(j=0; scanf("%d", &avail[i]);
    }
    for (i=0, i<n; i++)
    {
       for(j=0, j<m, j++)
          need[i][j]= req[i][j]- alloc[i][j];
```



```
for (i=0; i<m; i++)
    flag[i]=1;
k=1;
while (k)
{
  k=0;
  for (i=0; i<n; i++)
  {
    if (flag[i]);
    {
      c=0;
      for(j=0; j<m; j++)
      {
        if (need[i][j]<=avail[j])
          c++;
      }
      if (c==m)
      {
        printf("Resources can be allocated to
                process %d and available resources are
        for (j=0; j<m; j++)
            printf("%d", avail[j]);
        for (j=0; j<m; j++)
        {
          avail[j]=avail[j]+alloc[i][j];
          alloc[i][j]=0;
        }
        flag[i]=0;
        count++;
      }
    }
  }
```



```
    for(i=0; i<n; i++)
        prev[i]=flag[i];
  }
  if (count==n)
      printf ("No deadlock");
  else
      printf ("Deadlock detected");
}
```

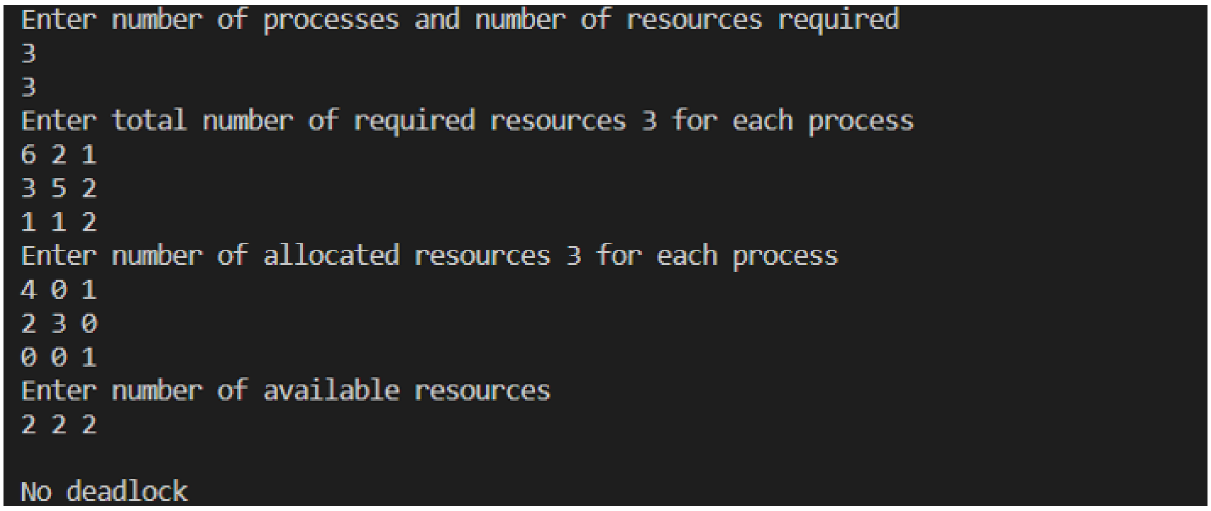
Output:

```
Enter no of processes and resources required :
Enter resources required
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Enter no. of allocated resources
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2
Enter number of allocated resources.
3 3 2
No deadlock.
```

10/10

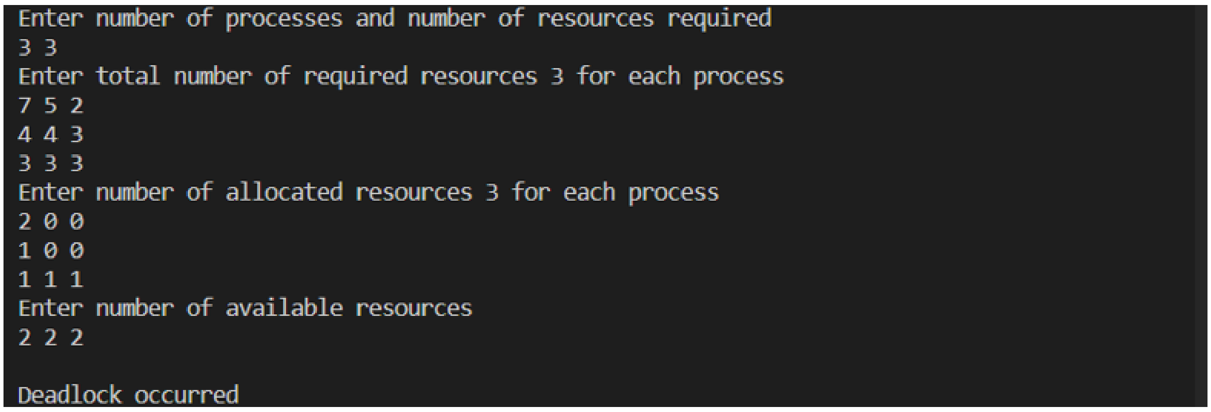
22/8/23

OUTPUT :



```
Enter number of processes and number of resources required
3
3
Enter total number of required resources 3 for each process
6 2 1
3 5 2
1 1 2
Enter number of allocated resources 3 for each process
4 0 1
2 3 0
0 0 1
Enter number of available resources
2 2 2

No deadlock
```



```
Enter number of processes and number of resources required
3 3
Enter total number of required resources 3 for each process
7 5 2
4 4 3
3 3 3
Enter number of allocated resources 3 for each process
2 0 0
1 0 0
1 1 1
Enter number of available resources
2 2 2

Deadlock occurred
```