Ex. No.: 9
Date: 05-04-25

DEADLOCK AVOIDANCE

Aim:

To find out a safe sequence using Banker's algorithm for deadlock avoidance.

Algorithm:

1. Initialize work=available and finish[i]=false for all values of i
2. Find an i such that both:
finish[i]=false and Need$_i$<= work
3. If no such i exists go to step 6
4. Compute work=work+allocationi
5. Assign finish[i] to true and go to step 2
6. If finish[i]==true for all i, then print safe sequence
7. Else print there is no safe sequence

Program Code:

```
#include <stdio.h>
#include <stdbool.h>
int main() {
    int m, n;
    printf ("Enter the number of resources and processes \n");
    scanf ("%d %d", &m, &n);
    int max[n][m];
    int allocation[n][m];
    printf ("\n Enter the values for max array: \n");
    for(int i=0; i<n; i++)
     { for(int j=0; j<m; j++)
        { scanf ("%d", &max[i][j]);
     }
    printf("\n Enter the values for allocation array: \n");
    for(int i=0; i<n; i++)
      { for(int j=0; j<m; j++)
         { scanf("%d", &allocation[i][j]);
         }
      }
```

```
{
  if (need[i][j] > work[j])
   {
      F = False;
      break;
   }
}
if (F)
{
   for (int j=0; j<m; j++)
        {
          work[j] += allocation[i][j];
        }
       finish[i] = true;
       s[k] = i;
       k++;
    }
  }
}}
for(int i=0;i<n-1;i++)
   {printf("p %d ->", s[i]);
   }
   printf("p %d ", s[n-1]);
  }
```

Input:

Enter the number of resources and process

3

5

Enter value for max array

7 5 3
3 2 2
9 0 2
2 2 2
4 3 3

Enter value for available array

3 3 2

Enter value for allocation array

0 1 0
2 0 0
3 0 2
2 1 1
0 0 2

Max	A	B	C
P_0	7	5	3
P_1	3	2	2
P_2	9	0	2
P_3	2	2	2
P_4	4	3	3

Allocation	A	B	C
P_0	0	1	0
P_1	2	0	0
P_2	3	0	2
P_3	2	1	1
P_4	0	0	2

Need

	A	B	C
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

Sample Output:

The SAFE Sequence is
P1 -> P3 -> P4 -> P0 -> P2

O/P

need mat

7 4 3
1 2 2
6 0 0
0 1 1
4 3 1

safe sequence is $P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_0 \rightarrow P_2$

Result:

Thus the C program for deadlock avoidance is successfully executed.