Ex. No.: 9
Date:

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>
# define MAX_PROCESSES 5
# define MAX_RESOURCES 3

bool is safe (int process[], int avail[], int max[, MAX_PROCESS]

        int allot[][MAX_PROCESSES][MAX_RESOURCES]
        bool finish[MAX_PROCESSES] = [false];
        int work[MAX_RESOURCES];
        for (int i=0; i<n; i++){
                for (int j=0; j<m; j++){
                        need[i][j] = max[i][j] - allot[i][j];
                }
        for (int i=0; i<m; i++){
                work[i] = avail[i];
        int safe sequence [MAX_PROCESSES];
        int count=0;
```

```
while (count<n){
    bool found = false;
    for (int i=0; i<n; i++){
        if (!finished[i]{
            int j;
            for (j=0; j<m; j++){
                if (need[i][j] > Work[j])
                    break;
            }
            if (j==m){
                for ( int k=0; k<m; k++){
                    work[k] -= allot[i][k];
            }
            finish[i] = true;
            Safe sequence[count++] = i;
            found = True;
            break;
            }
        }
    }
    Printf("Safe Sequence");
    for (int i=0; i<n; i++){
        Print("P %d", Safe Sequence[i]);
    }
    Printf("\n");
    return true;
}
int main(){
    int process[MAX_PROCESSES];
    int avail[MAX_RESOURCES];
    Printf("Enter available resources (ABC):");
    for (i=0; i<MAX_RESOURCES; i++){
        Scanf("%d", &avail[i]);
}
```

OUTPUT:

Enter no. of. process + resource : 5 3

Enter alloc of resource of all process

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

Enter max resource required :

7 5 3
3 2 2
9 0 2
4 3 2
5 8 3

Enter available resource : 3 3 2

Need Resource matrix :

7 4 3
1 2 2
6 0 0
2 1 1
5 3 1

Sample Output:

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

Available resource after Completion : 10 5 7

Safe sequence : $P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_0 \rightarrow P_2$

Result:

Hence Safe Sequence is obtained for dead lock avoidance using bankers algorithm.

PDF Scanner ACE Scanner