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>
int main(){
    int p, c, count =0, i, j;
    printf ("Enter the no.of processo and resources\n")
    scanf ("%d %d", &p, &c);
  int alc [p] [c], max [p] [c], need [p] [c], safe [p],
                        available [c], done [5], terminated=0;

  for ( i=0; i<p; i++){
            for ( j=0; j<c; j++){
                Scanf ("%d", & max [i] [j]);
      }
  }

  Printf ("Enter the available resources");
    for ( i=0; i<c; i++){
        scanf ("%d", &available [i]);

printf ("\n need resource matrix are \n");
    ~~for (1~~
  for( i=0; i<P; i++){
          done [i]=0;
    }
while ( count <P){
        for ( i=0; i<P; i++){
            if (done [i] = =0){
                for (j=0; j<c; j++){
                    if(need [i][j] > available [j])
                    break;
            }
      if (j = = c){
          safe [count ] = i;
          done [i] =1;
          for( j = 0; i <c; j++ ){
                available [j] += alc [i] [j];
            }
            count ++;
            terminate =0;
          } else {
                terminate ++;
            }
        }
    }
  if ( terminate = =(P-1)){
        printf("Safe Sequence does not exist");
        break;
    }
  }
```

```
if ( terminate != (P-1) ){
    printf ("\n available resources after Completion \n")
    for ( i= 0; i<c ; i++) {
        printf ("%d \t", available [i]);
    }
    printf ("\n Safe Sequence Are \n");
    for ( i=0; i< P; i++){
        printf (" P%d \t ", safe [i]);
    } }
    return 0;
}
```

Sample Output:

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

The safe sequence is:

$P_2 \Rightarrow P_1 \Rightarrow P_0$

[illegible]

Result:

Hence the deadlock avoidance using Bankers Algorithm has been implemented and executed successfully.