Ex No.: 9 BANKER DEADLOCK AVOIDANCE ALGORITHM

Date : 03.04.2025

Aim :

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

Code:

```
#include <stdio.h>
#define MAX_P 10
#define MAX_R 10
int main() {
    int avail[MAX_R];
    int max[MAX_P][MAX_R];
    int alloc[MAX_P][MAX_R];
    int need[MAX_P][MAX_R];
    int done[MAX_P] = {0};
    int work[MAX_R];
    int safe_seq[MAX_P];
    int p, r, i, j, cnt = 0;
    printf("Enter number of processes and resources: ");
    scanf("%d %d", &p, &r);
    printf("Enter available resources: ");
    for (i = 0; i < r; i++) {
        scanf("%d", &avail[i]);
        work[i] = avail[i];
    }
```

```
printf("Enter max matrix:\n");
for (i = 0; i < p; i++) {
    printf("For process P%d: ", i);
    for (j = 0; j < r; j++)
        scanf("%d", &max[i][j]);
}
printf("Enter allocation matrix:\n");
for (i = 0; i < p; i++) {
    printf("For process P%d: ", i);
    for (j = 0; j < r; j++) {
        scanf("%d", &alloc[i][j]);
        need[i][j] = max[i][j] - alloc[i][j];
    }
}
while (cnt < p) {
    int found = 0;
    for (i = 0; i < p; i++) {
        if (!done[i]) {
            int can_run = 1;
            for (j = 0; j < r; j++)
                if (need[i][j] > work[j]) can_run = 0;
            if (can_run) {
                for (j = 0; j < r; j++)
                    work[j] += alloc[i][j];
                safe_seq[cnt++] = i;
```

```
                done[i] = 1;
                found = 1;
            }
        }
    }
    if (!found) break;
  }
  if (cnt == p) {
    printf("Safe sequence exists: ");
    for (i = 0; i < p; i++)
      printf("P%d ", safe_seq[i]);
  } else {
    printf("No safe sequence exists (deadlock possible)");
  }
  return 0;
}
```

Output:

```
Enter number of processes and resources: 5 3
Enter available resources: 3 3 2
Enter max matrix:
For process P0: 7 5 3
For process P1: 3 2 2
For process P2: 9 0 2
For process P3: 2 2 2
For process P4: 4 3 3
Enter allocation matrix:
For process P0: 0 1 0
For process P1: 2 0 0
For process P2: 3 0 2
For process P3: 2 1 1
For process P4: 0 0 2
Safe sequence exists: P1 P3 P4 P0 P2
```

Result:

Thus the program to find out the safe sequence using Banker's algorithm for deadlock avoidance has been executed successfully.