# Topic 17: 17. Illustrate the deadlock avoidance concept by simulating Banker’s algorithm with C.

#include <stdio.h>  
#define P 5  
#define R 3  
  
int main() {  
 int alloc[P][R] = {{0, 1, 0}, {2, 0, 0}, {3, 0, 2}, {2, 1, 1}, {0, 0, 2}};  
 int max[P][R] = {{7, 5, 3}, {3, 2, 2}, {9, 0, 2}, {2, 2, 2}, {4, 3, 3}};  
 int avail[R] = {3, 3, 2};  
 int f[P], ans[P], ind = 0;  
 int need[P][R];  
 for (int i = 0; i < P; i++) {  
 f[i] = 0;  
 for (int j = 0; j < R; j++) {  
 need[i][j] = max[i][j] - alloc[i][j];  
 }  
 }  
 for (int k = 0; k < P; k++) {  
 for (int i = 0; i < P; i++) {  
 if (!f[i]) {  
 int flag = 1;  
 for (int j = 0; j < R; j++) {  
 if (need[i][j] > avail[j]) {  
 flag = 0;  
 break;  
 }  
 }  
 if (flag) {  
 for (int y = 0; y < R; y++) avail[y] += alloc[i][y];  
 ans[ind++] = i;  
 f[i] = 1;  
 }  
 }  
 }  
 }  
 return 0;  
}