BANKER'S ALGORITHM

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

```
                f[i] = 1;
            }
        }
    }
}
int flag = 1;
for (i = 0; i < n; i++) {
    if (f[i] == 0) {
        flag = 0;
        printf("System is not safe\n");
        break;
    }
}
if (flag == 1) {
    printf("System is safe. Safe sequence is:\n");
    for (i = 0; i < n - 1; i++) printf("P%d -> ", ans[i]);
    printf("P%d\n", ans[n - 1]);
}
return 0;
}
```