#include <stdio.h>

#include <stdbool.h>

#define P 5 // Number of processes

#define R 3 // Number of resources

int main() {

int i, j, k, p, count = 0;

int processes[] = {0, 1, 2, 3, 4};

int avail[] = {3, 3, 2};

int max[P][R] = {{7, 5, 3}, {3, 2, 2}, {9, 0, 2}, {2, 2, 2}, {4, 3, 3}};

int allot[P][R] = {{0, 1, 0}, {2, 0, 0}, {3, 0, 2}, {2, 1, 1}, {0, 0, 2}};

int work[R], finish[P] = {0}, safeSeq[P];

int req1[] = {1, 0, 2}; // Process 1 request

for (i = 0; i < R; i++) work[i] = avail[i];

for (i = 0; i < R; i++) {

if (req1[i] > max[1][i] - allot[1][i] || req1[i] > avail[i]) {

printf("Request cannot be granted.\n");

return 0;

}

}

for (i = 0; i < R; i++) avail[i] -= req1[i], allot[1][i] += req1[i];

while (count < P) {

bool found = false;

for (p = 0; p < P; p++) {

if (!finish[p]) {

for (j = 0; j < R; j++)

if (max[p][j] - allot[p][j] > work[j]) break;

if (j == R) {

for (k = 0; k < R; k++) work[k] += allot[p][k];

safeSeq[count++] = p;

finish[p] = true;

found = true;

}

}

}

if (!found) {

printf("System is not in a safe state\n");

return 0;

}

}

printf("System is in a safe state.\nSafe sequence is: ");

for (i = 0; i < P; i++) printf("%d ", safeSeq[i]);

printf("\n");

return 0;

}

