**import** java.util.\*;

**class** Osproject

{

**static** **int** P = 3;

**static** **int** R = 3;

**static** **void** calculateNeed(**int** need[][], **int** maxm[][],

**int** allot[][])

{

**for** (**int** i = 0 ; i < P ; i++)

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

need[i][j] = maxm[i][j] - allot[i][j];

}

**static** **boolean** isSafe(**int** processes[], **int** avail[], **int** maxm[][],

**int** allot[][])

{

**int** [][]need = **new** **int**[P][R];

calculateNeed(need, maxm, allot);

**boolean** []finish = **new** **boolean**[P];

**int** []safeSeq = **new** **int**[P];

**int** []work = **new** **int**[R];

**for** (**int** i = 0; i < R ; i++)

work[i] = avail[i];

**int** count = 0;

**while** (count < P)

{

**boolean** found = **false**;

**for** (**int** p = 0; p < P; p++)

{

**if** (finish[p] == **false**)

{

**int** j;

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

**if** (need[p][j] > work[j])

**Break;**

**if** (j == R)

{

**for** (**int** k = 0 ; k < R ; k++)

work[k] += allot[p][k];

safeSeq[count++] = p;

finish[p] = **true**;

found = **true**;

}

}

}

**if** (found == **false**)

{

System.out.print("System is not in safe state");

**return** **false**;

}

}

System.out.print("System is in safe state.\nSafe"

+" sequence is: ");

**for** (**int** i = 0; i < P ; i++)

System.out.print(safeSeq[i] + " ");

**return** **true**;

}

**public** **static** **void** main(String[] args)

{

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

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

**int** maxm[][] = {{8, 4, 3},

{6, 2, 2},

{4, 3, 3}};

**int** allot[][] = {{0, 0,3},

{3, 2, 0},

{2, 1, 1}};

isSafe(processes, avail, maxm, allot);

}

}