QB: It is required to maintain and process the status of total 9 resources. The status value is to be stored in an integer array of dimension 3x3. The valid status of a resource can be one of the 2followings:free: indicated by integer value 0 occupied: indicated by integer value 1

inaccessible: indicated by integer value 2

Declare a class called ResourcesStatus, having data member called statusRef, referring to a two dimensional array (3x3) of integers to be used to refer to the above mentioned status values. Define a member method called processStausCount that counts and displays total number of free resources, total number of occupied resources and total number of inaccessible resources. The exception to be raised and handled if total number of occupied resources exceeds total number of free resources. The handler marks status of all inaccessible resources as free. Accept initial status values from command line arguments and initialize the array. Raise and handle user defined exception if invalid status value given

```
class InvalidEntry extends RuntimeException {
    InvalidEntry() {
        System.out.println("enter number 0,1,2 only");
    }
}

class ExceedNumber extends RuntimeException {
    ExceedNumber() {
        System.out.println("Occupied is above free");
    }
}

class ResourseStatus {
    int seat[][] = new int[3][3];
    int free, occupied, inaccessible;

    ResourseStatus(String args[]) {
        int n = 0;
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                seat[i][j] = Integer.parseInt(args[n]);
                try {
                    if (seat[i][j] > 2 || seat[i][j] < 0) {
                        throw new InvalidEntry();
                    }
                } catch (Exception e) {
```

```
                    System.out.println(seat[i][j] + "is Invalid Number");
                }
                n++;
            }
        }
    }

    public void processStatusCount() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                if (seat[i][j] == 0)
                    free++;
                else if (seat[i][j] == 1)
                    occupied++;
                else if (seat[i][j] == 2)
                    inaccessible++;
                else
                    continue;
            }
        }
        try {
            if (occupied > free) {
                throw new ExceedNumber();
            }
        } catch (Exception e) {
            System.out.println("neet to create more free seats");
            free = free + inaccessible;
            inaccessible = 0;
        }
        System.out.println("Total free seats = " + free);
        System.out.println("Total occupied seats = " + occupied);
        System.out.println("Total inccessible seats = " + inaccessible);
    }
}

public class P3 {
    public static void main(String[] args) {
        ResourseStatus r1 = new ResourseStatus(args);
        r1.processStatusCount();
    }
}
```