The Music Department uses a computer program to keep track of the seats sold for their spring concert. One arrangement of the chairs for the concert has 24 rows with 35 seats in each row. To keep track of the seats sold they use a two dimensional array which is declared as follows:

**boolean** [][] seatsSold = **new boolean**[24][35];

Originally all entries in the array would be false (seat not sold). As seats are sold, the corresponding array element is changed to true. For example, if you sold seat number 7 in the third row, you would mark this seat as sold with the statement:

seatsSold[2][6] = **true**;

In this case we used 2 and 6 because the array is zero based. If you want, you can use a 25 by 36 array and ignore the zero row and column.

1. Write a method called clearSeats() that resets all of seats in the seatsSold array to not sold (false). To help get you started, use the following method heading:

**static void** clearSeats(**boolean** [][] seatsSold)

1. Write a complete Java method called bestRowAvailable() that finds and returns the lowest row number (row closest to the front) that has the number of seats that you are requesting beside each other in a row. For example if you were looking for 5 tickets in the three rows shown below where T is a sold seat and F is an unsold seat, row 2 is the lowest row with 5 seats beside each other. If there are no rows with the number of seats you are requesting together, the method should return –1.

Front of Theatre

Row 1 TTTFFTTFFTTFTTFFTTTTFFTTFFFTTFFTTFF

Row 2 TTTTTFFTTTT**FFFFF**TTFFTTTTTTTTTTTTTTT

Row 3 TTTTTFFFFFFFFTTTTFFFFFTTFFTTTTTTTTT

This method should have 2 parameters: the seatsSold array and the number of tickets you are looking for. To help get you started, the method heading is given below:

**static int** bestRowAvailable (**boolean** [][] seatsSold,

**int** noOfTickets)