**Write a pseudocode to solve the skyline problem**

Define: the first element of array start at index 0

Define: Read() mean get input from the problem

Define: new DataType[size] mean declare array

Define: Building as an structure in this problem and it has own attribute which is

integer : lefPosition

integer : rightPosition

integer : high

Define: Building(integer : L, integer : R, integer : H) :

lefPosition = L

rightPosition = R

high = H

Define: getHigh():

return high

Define: getRightPosition():

return rightPosition

Define: getLeftPosition():

return leftPosition

Start

t <- Read()

city <- new Building[t]

# input part

For Let i <- 0 To t-1 Step i By 1 Then

L <- Read()

R <- Read()

H <- Read()

city[i] <- Building(L,R,H)

EndFor

# process part

cityBouder <- city[0].getRightPosition()

For Let i <- 0 To t-1 Step i By 1 Then

If cityBouder < city[i].getRightPosition() Then

cityBouder <- city[i].getRightPosition()

EndIf

EndIf

For Let i <- 0 To t-1 Step i By 1 Then

For Let j <- city[i].getLeftPosition() To city[i].getRightPosition()-1 Step j By 1 Then

If newcity[j] < city[i].getHigh() Then

newcity[j] <- city[i].getHigh()

EndIf

Endfor

Endfor

newcity <- new Integer[cityBouder+2]

#output part

oldHigh <- 0

For Let i <- 0 To cityBouder Step i By 1 Then

If oldHigh != newcity[i] Then

Display i " " newcity[i] " "

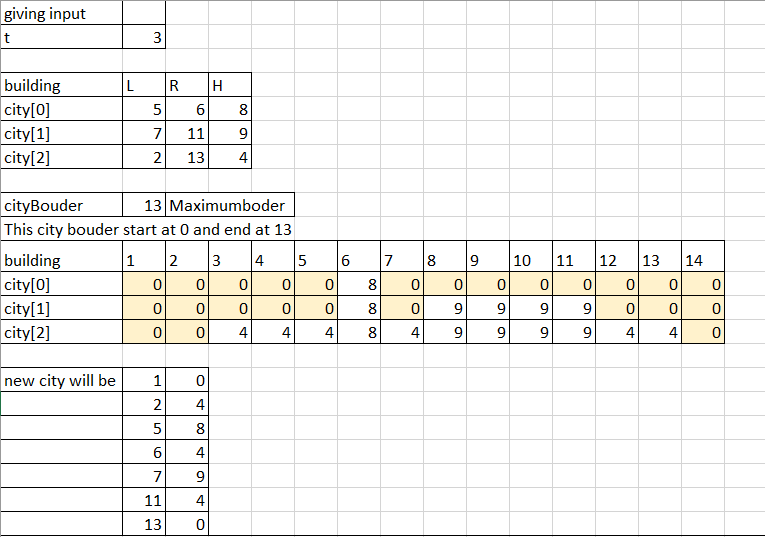
oldHigh <- newcity[i]

EndIf

EndFor

End

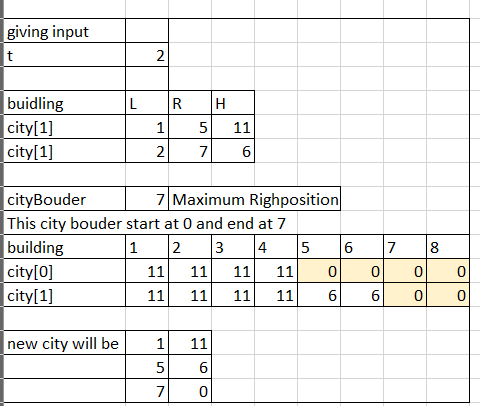
**Show that your pseudocode correct**



Or you can arrange to (2,4,5,8,6,4,7,9,11,4,13,0)

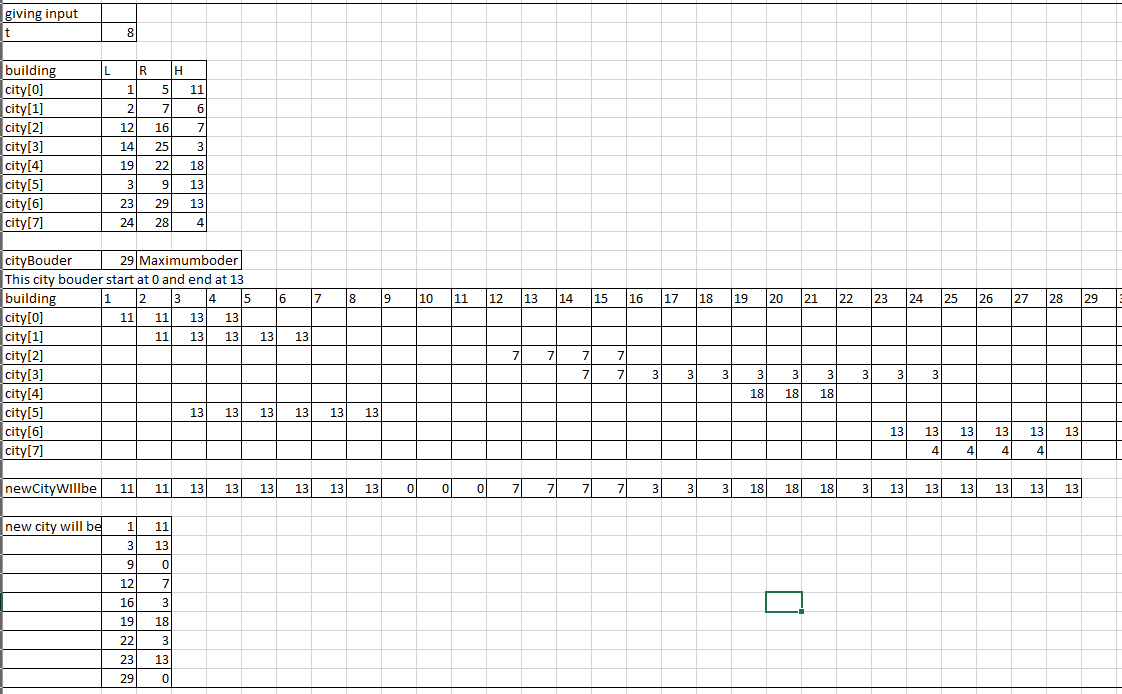
**Show that you understand the problem by write all possible case + data of the problem instances**

**Small input**

**2 (1,5,11),(2,7,6)**

**Output will be (1,11,5,6,7,0)**

**Big input**

**8 (1,5,11),(2,7,6),(12,16,7),(14,25,3),(19,22,18),(3,9,13),(23,29,13),(24,28,4)**

**Output will be**

**(1,11,3,13,9,0,12,7,16,3,19,18,22,3,23,13,29,0)**