Write a pseudocode to solve the skyline problem

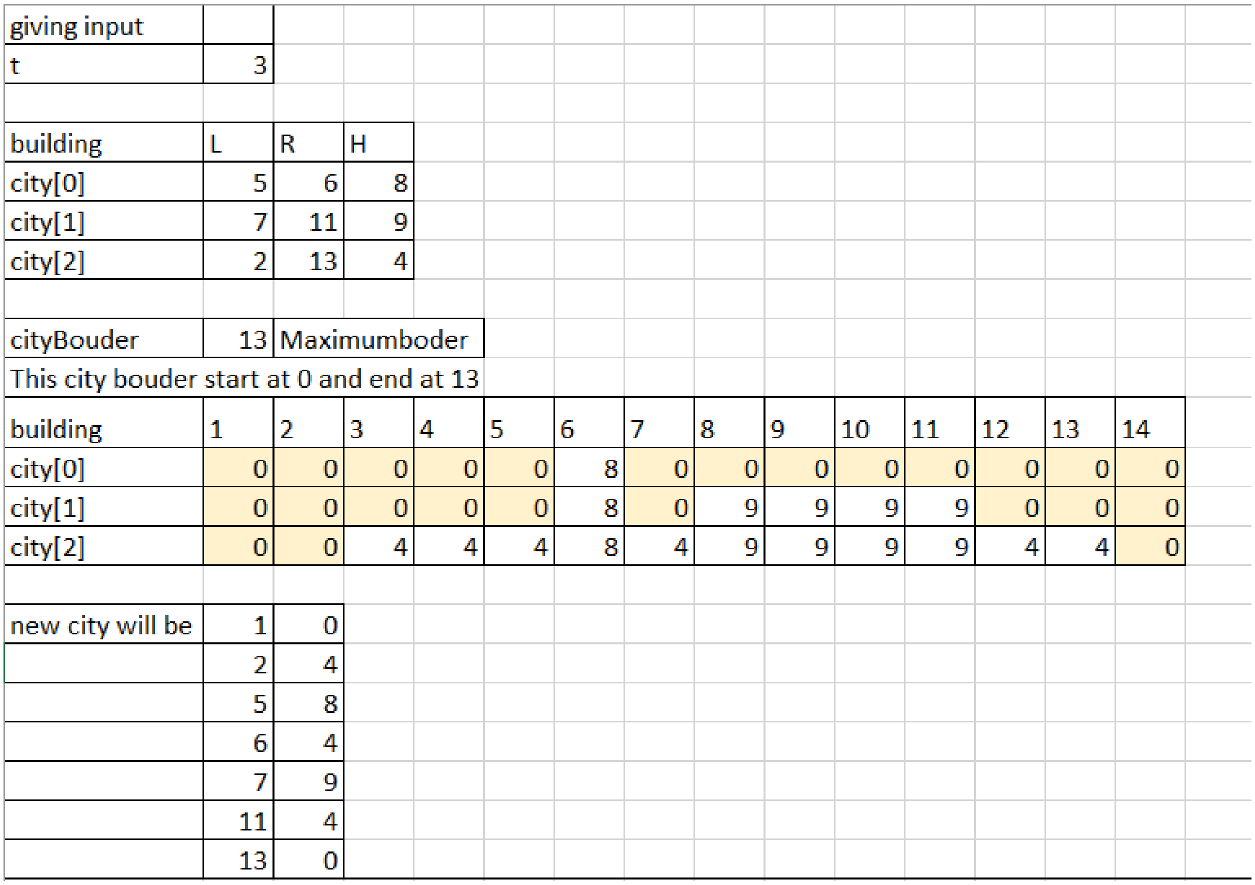
```
Define: the first element of array start at index 0

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



giving input	
t	3

building	L	R	H
city[0]	5	6	8
city[1]	7	11	9
city[2]	2	13	4

cityBouder	13	Maximumboder

This city bouder start at 0 and end at 13

building	1	2	3	4	5	6	7	8	9	10	11	12	13	14
city[0]	0	0	0	0	0	8	0	0	0	0	0	0	0	0
city[1]	0	0	0	0	0	8	0	9	9	9	9	0	0	0
city[2]	0	0	4	4	4	8	4	9	9	9	9	4	4	0

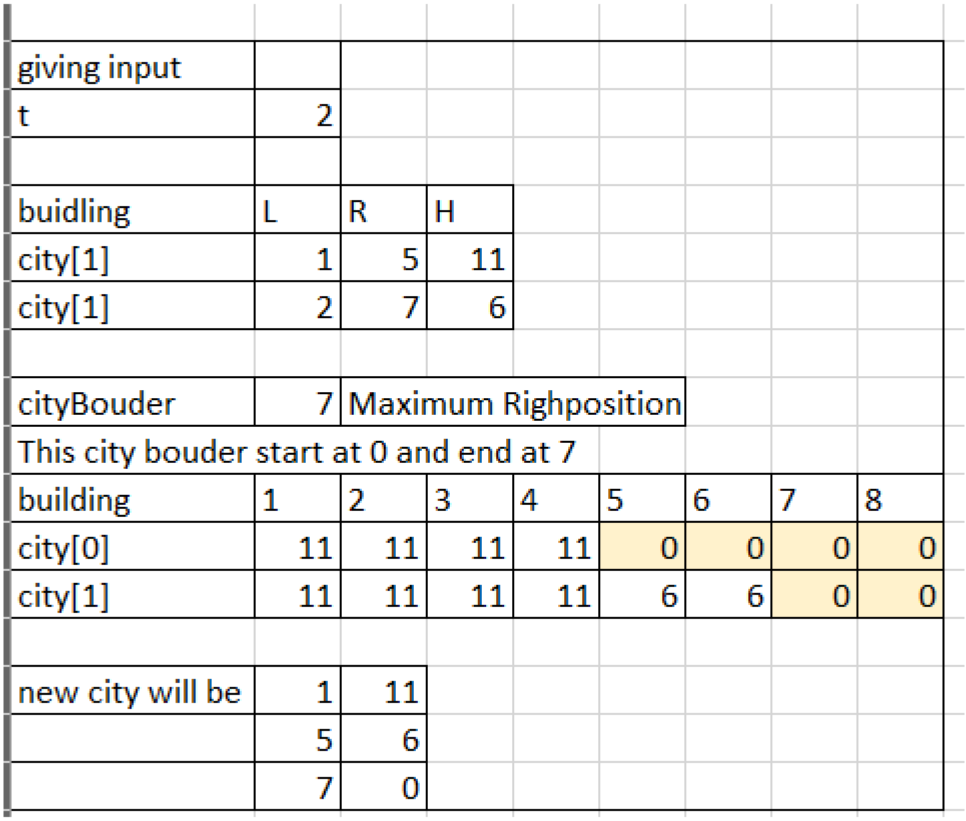
new city will be	1	0
	2	4
	5	8
	6	4
	7	9
	11	4
	13	0

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)

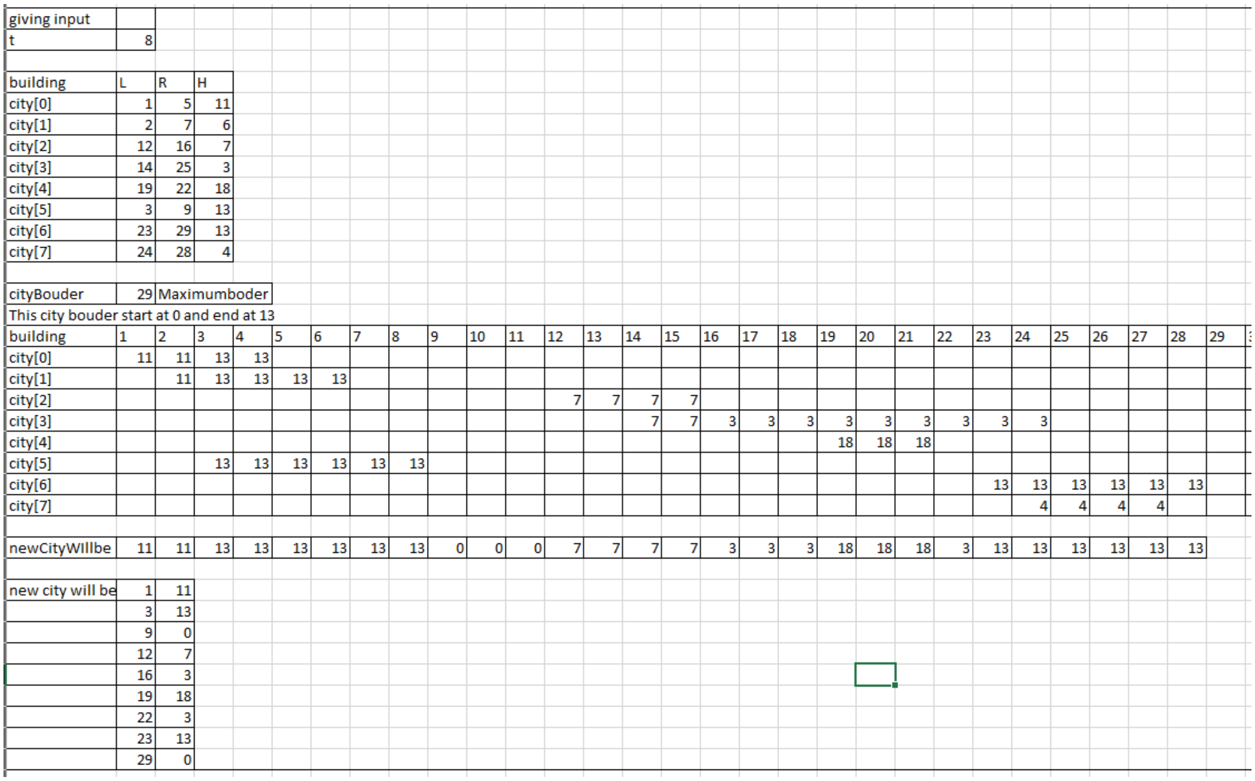


giving input								
t	2							
buidling	L	R	H					
city[1]	1	5	11					
city[1]	2	7	6					
cityBouder	7	Maximum Righposition						
This city bouder start at 0 and end at 7								
building	1	2	3	4	5	6	7	8
city[0]	11	11	11	11	0	0	0	0
city[1]	11	11	11	11	6	6	0	0
new city will be	1	11						
	5	6						
	7	0						

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)



giving input	
t	8

building	L	R	H
city[0]	1	5	11
city[1]	2	7	6
city[2]	12	16	7
city[3]	14	25	3
city[4]	19	22	18
city[5]	3	9	13
city[6]	23	29	13
city[7]	24	28	4

cityBouder	29	Maximumboder

This city bouder start at 0 and end at 13

building	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	3
city[0]	11	11	13	13																										
city[1]		11	13	13	13	13																								
city[2]												7	7	7	7															
city[3]														7	7	3	3	3	3	3	3	3	3	3						
city[4]																			18	18	18									
city[5]			13	13	13	13	13	13																						
city[6]																							13	13	13	13	13	13		
city[7]																								4	4	4	4			

newCityWIllbe	11	11	13	13	13	13	13	13	0	0	0	7	7	7	7	3	3	3	18	18	18	3	13	13	13	13	13	13

new city will be	1	11
	3	13
	9	0
	12	7
	16	3
	19	18
	22	3
	23	13
	29	0

Output will be

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