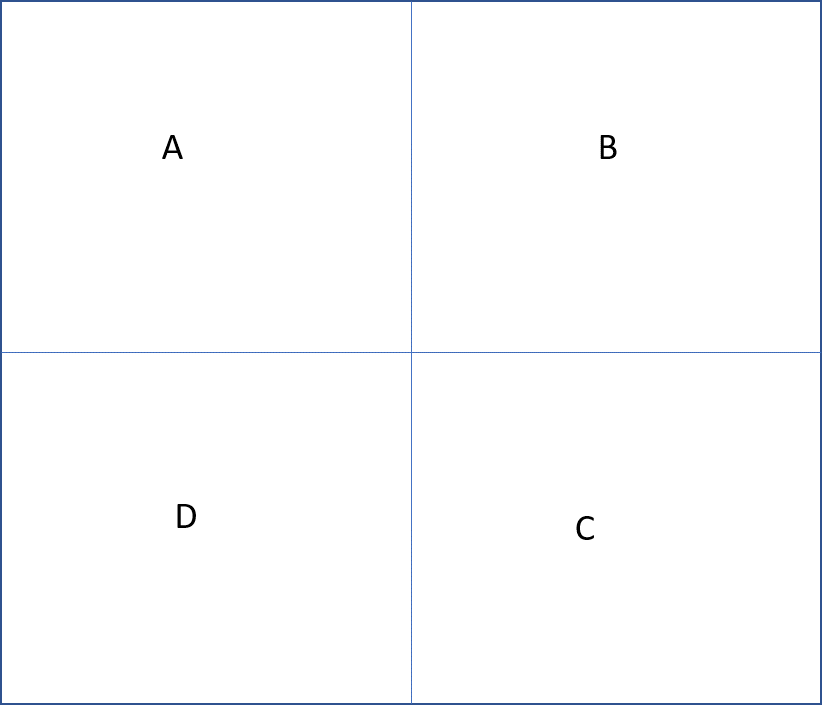
README FILE

**To run the program :**run the : *main.py* file with the path to file you want to try.  
full command (it works this way on my machine) :   
python main.py PATH\_TO\_FILE

**Elaboration about the algorithm which I used to solve the problem:**

I started with the case of even-edge square board.  
I divided the game board to four parts :  


When I read the boxes and hunters coordinates I counted the number of boxes and hunters in each part, then I concluded the number of free seats in each part.  
Then, I use the follow equations :  
Fix for each part A,B,C,D

Now, according to the rules I get :  
Then, I tried to find maximal X values such that :

For the case of odd-edge game boards :   
I created different partition to the game board -  
A screenshot of a cell phone

Description automatically generated

The idea is that the only difference between the even and odd game board is the row or column in the middle.  
So I set the row/column as a different part of the board (  
And then I have to balance the parts A,B,C,D like before and I have to balance E against G and H against F.  
The part I In the center is just the center point of the board (size 1\*1).